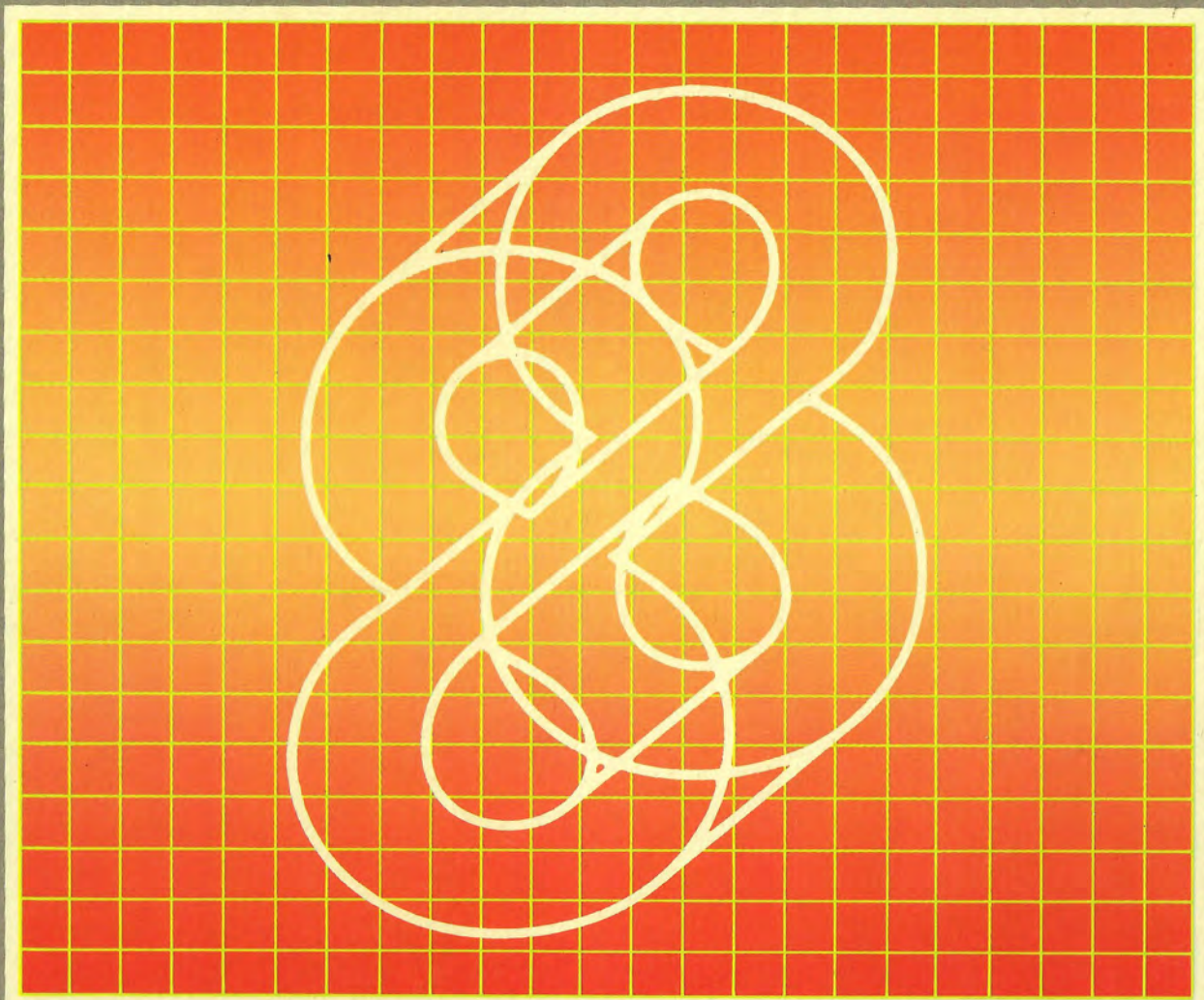


BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



PON A TRABAJAR A TU ORDENADOR

HAZ COSAS UTILES CON EL

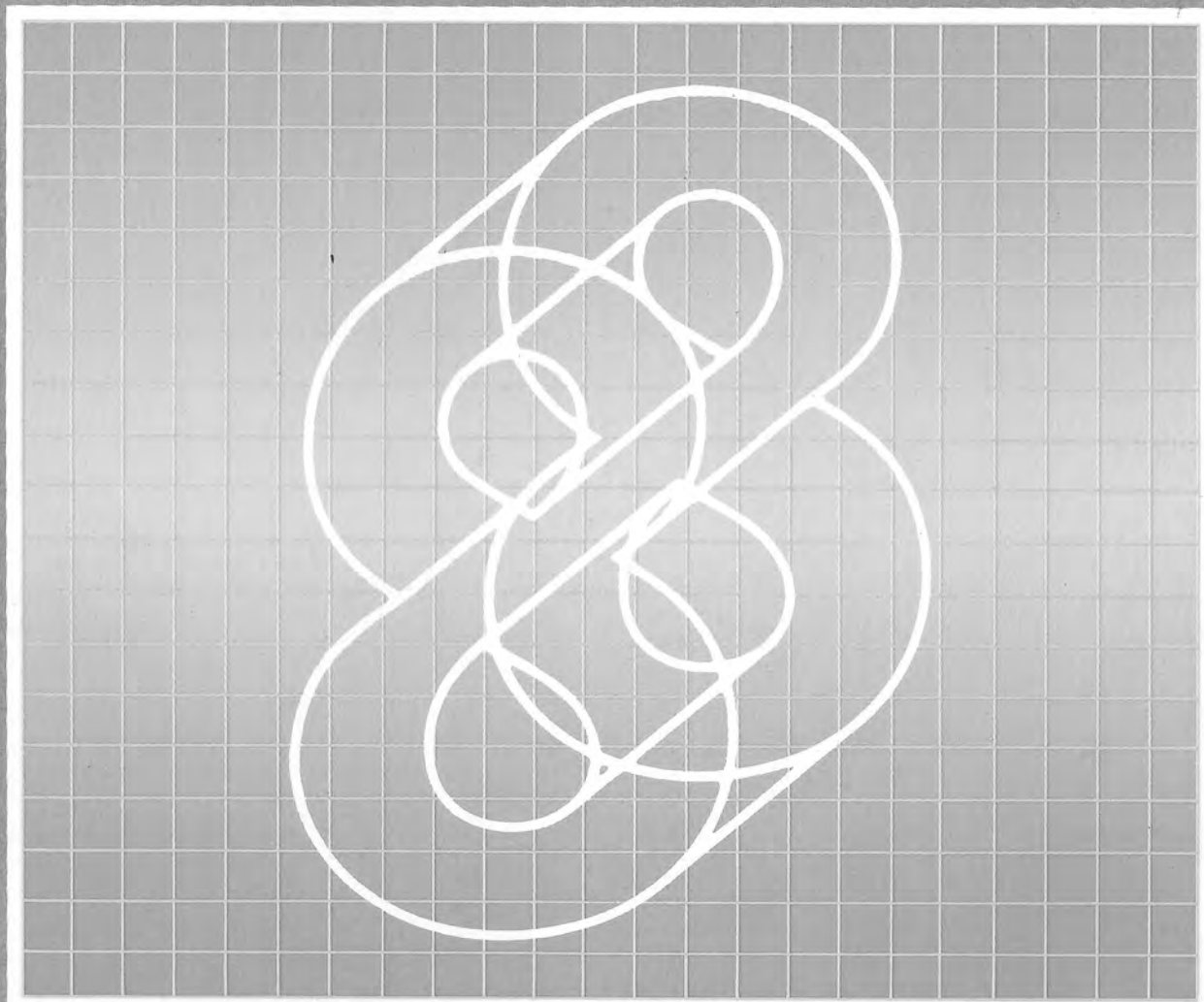
EDICIONES SIGLO CULTURAL

BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



PON A TRABAJAR A TU ORDENADOR

HAZ COSAS UTILES CON EL

EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S. A.

Director-editor:

RICARDO ESPAÑOL CRESPO.

Gerente:

ANTONIO G. CUERPO.

Directora de producción:

MARIA LUISA SUAREZ PEREZ.

Director de la colección:

JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño:

BRAVO-LOFISH.

Maquetación:

D. S. M.

Dibujos:

JOSE OCHOA

Tomo 8. «Experiencias prácticas en logo», Carlos Albert, Técnico de Informática; Adoración Llena Jubero, Técnico de Informática. «Manejo de sprites y elementos gráficos», «Trucos y rutinas básicas», Francisco Morales. Técnico de Informática. «Aprende con el ordenador», AULA DE INFORMATICA APLICADA: Soledad Tamariz-Martel, Diplomada en Telecomunicación; Francisco Blanca, Diplomado en Telecomunicación; Alejandro Marcos, Licenciado en Química; Fernando Suero, Diplomado en Telecomunicación. «El Taller del Hardware», Carlos Rey, Ingeniero Industrial. «Pequeña historia de la Informática», «Temas monográficos de vanguardia», «Terminología», «Vocabulario de Informática», Blanca Arbizu, Técnico de Informática.

Ediciones Siglo Cultural, S. A.

Dirección, redacción y administración:

Sor Angela de la Cruz, 24-7.º G. Teléf. 279 40 36. 28020 Madrid.

Publicidad:

Gofar Publicidad, S. A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:

COEDIS, S. A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.

Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:

CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.

Buenos Aires - 1.290. Argentina.

Todos los derechos reservados. Este libro no puede ser, en parte o totalmente, reproducido, memorizado en sistemas de archivo, o transmitido en cualquier forma o medio, electrónico, mecánico, fotocopia o cualquier otro, sin la previa autorización del editor.

ISBN del tomo: 84-7688-065-0

ISBN de la obra: 84-7688-047-2

Fotocomposición:

ARTECOMP, S. A. Albarracín, 50. 28037 Madrid.

Imprime:

MATEU CROMO. PINTO (Madrid).

© Ediciones Siglo Cultural, S. A., 1986

Depósito legal: M-43.185-1986

Printed in Spain - Impreso en España.

Suscripciones y números atrasados:

Ediciones Siglo Cultural, S. A.

Sor Angela de la Cruz, 24-7.º G. Teléf. 279 40 36. 28020 Madrid.

Marzo, 1987

INDICE

■ EXPERIENCIA Y PRACTICAS EN LOGO	5
■ MANEJO DE SPRITES Y ELEMENTOS GRAFICOS	16
■ TRUCOS Y RUTINAS BASICAS	24
■ EL TALLER DEL HARDWARE	32
■ APRENDER CON EL ORDENADOR	40
■ PEQUEÑA HISTORIA DE LA INFORMATICA	50
■ TEMAS MONOGRAFICOS DE VANGUARDIA	54
■ TERMINOLOGIA	58
■ VOCABULARIO DE INFORMATICA	60

EXPERIENCIA Y PRACTICAS EN LOGO

La pantalla: escala



Al dibujar un cuadrado o un triángulo has observado que los cuadrados no parecen cuadrados y que los círculos no parecen redondos, pregúntale al LOGO la escala actual de la pantalla gráfica.

? ESCRIBE ESCALA

Dependiendo de la versión LOGO con la que trabajes, obtendrás lo siguiente:

- Dos números, el primero corresponde a la escala del eje X y el segundo a la del eje Y. Representan el número de puntos o centésimas de punto que avanza la Tortuga al dar un paso.

- Un solo número indica la proporción que existe entre los pasos horizontales y verticales de la Tortuga.

Puedes cambiar la longitud de paso de la Tortuga. Dependiendo de la versión será:

? PONESCALA [X Y]

donde X e Y son los números que fijarán la nueva escala de la pantalla; ambos deben estar comprendidos entre:

1 y 200, en algunas versiones para los PC-Compatibles.

0 y 255, en los MSX.

o bien:

? PONESCALA n

n determinará la nueva proporción entre el eje de la X y el de la Y.

Con el mismo procedimiento que dibujamos un círculo.

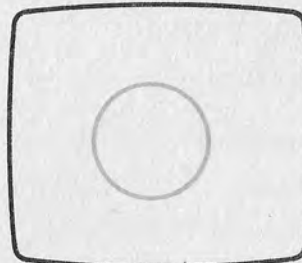


Fig. 1.

podemos dibujar una elipse.

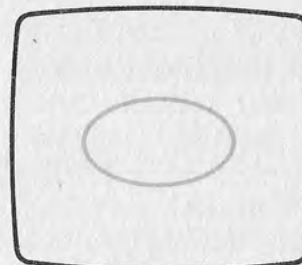


Fig. 2.

Compruébalo con este ejemplo que contiene dos procedimientos, variamos la forma de un mismo dibujo con sólo cambiar la escala.

El primero dibuja un círculo y el segundo varía la escala de la pantalla y llama al primero.

Antes de empezar a teclear el ejemplo anota la escala inicial de la pantalla.

Podemos cambiar la escala de la pantalla.

EXPERIENCIA Y PRACTICAS EN LOGO

? ESCRIBE ESCALA

Y ahora empieza con los procedimientos.

Primer procedimiento:

? PARA CIRCULO

- > BP
- > REPITE 36 [AV 4 GD 10]
- > FIN

Segundo procedimiento:

? PARA DIBUJOS

- > BP
- > PM
- > CIRCULO
- > PONESCALA [100 150]
- > ESCRIBE ESCALA
- > CIRCULO
- > PONESCALA [20 90]
- > ESCRIBE ESCALA
- > CIRCULO
- > PONESCALA [160 100]
- > ESCRIBE ESCALA
- > CIRCULO
- > FIN

Para ejecutarlo:

? DIBUJOS

En el caso de PONESCALA n, sustituye las siguientes órdenes en el procedimiento DIBUJOS:

PONESCALA [100 150] por PONESCALA 0.8
PONESCALA [20 90] por PONESCALA 0.3
PONESCALA [160 100] por PONESCALA 1

Cuando termines de probar lo que puedes llegar a hacer con esta orden, puedes reinicializar la escala de la pantalla dándole a la orden PONESCALA el valor que anteriormente has anotado.

? PONESCALA [x y]

o bien

? PONESCALA n

■ Guardar y cargar procedimientos III

A la hora de guardar en un soporte externo todo o parte del área de trabajo del ordenador, damos un nombre para definir el fi-

chero, en el que se van a guardar todos los procedimientos o únicamente los que especificamos.

Vimos que el nombre que se daba al fichero podía ser cualquiera y únicamente existían limitaciones en cuanto al número de caracteres que tenía.

No podemos utilizar un mismo nombre de fichero para guardar distintos procedimientos, ya que es lógico pensar que se grabaría sobre el mismo fichero borrándose lo que anteriormente hubiese grabado.

Si utilizamos disquete, y a la hora de guardar nuestros procedimientos damos un nombre al fichero igual a alguno de los que se encuentre en el disquete, el Logo nos avisa de que ya existe un fichero con ese nombre y que, por tanto, no podemos utilizarlo. Con sólo cambiar el nombre, podemos entonces guardar nuestros procedimientos.

Si utilizamos cinta de cassette, sí podemos usar el mismo nombre para dos ficheros diferentes, ya que se graban en zonas distintas de la cinta, siempre y cuando no tengamos la cinta posicionada en una zona en donde no haya nada anteriormente grabado. Si hubiese algo grabado en ella y grabásemos encima, todo lo anterior se borraría, permaneciendo lo último que hubiésemos grabado.

Por eso, comprueba siempre que no existe nada grabado en la zona de la cinta donde quieres grabar tu nueva información. Puedes comprobarlo escuchando la cinta. Si recibes algún sonido, significa que hay algo grabado en esa zona y si, por el contrario, no recibes sonido alguno, significará que esa zona está limpia y puedes guardar en ella lo que quieras.

También te puede servir de ayuda ir apuntando el número que aparece en el contador de vueltas de tu cassette, cuando comienzas a grabar y cuando terminas, poniéndolo previamente a cero.

Si trabajamos con disquete, puedes decirle al Logo que te dé información sobre su contenido.

Esto lo conseguimos gracias a la orden:

DIR

Esta orden no es exclusiva del Logo, ya

Ahorra memoria utilizando las abreviaturas de las órdenes

que también se utiliza en otros lenguajes de programación.

DIR es la abreviatura de DIRECTORIO y cuando lo introducimos, recibimos la lista de todos los ficheros que se encuentran en el disquete.

Si tienes un SPECTRUM que disponga de microdrive, esta información la puedes obtener con la orden:

CATALOGO

Esto puede ser de gran ayuda a la hora de dar un nombre a un nuevo fichero que deseamos guardar y no nos acordamos si lo hemos utilizado anteriormente.

Si queremos cargar un fichero determinado en la memoria del ordenador y no nos acordamos de su nombre, con esta orden podremos visualizar todos los que existen y recordar el que nos interesa.

El disquete tiene una capacidad limitada, por lo que si vamos a guardar un fichero y no existiese espacio en el disquete, recibiremos un mensaje comunicándonoslo.

Vimos que dependiendo del ordenador y de la versión de Logo con la que trabajábamos, podíamos guardar los procedimientos de dos formas, bien dando exclusivamente un nombre al fichero y guardándose bajo este nombre todos los procedimientos que hubiese en la memoria, o bien especificando tanto el nombre del fichero como los nombres procedimientos que queríamos guardar.

En el caso de los ordenadores que nos permite la primera forma, también es posible realizar la segunda.

En los ordenadores MSX podemos tanto dar exclusivamente un nombre del fichero, guardándose todos los procedimientos, como especificar los que queremos guardar. Esto lo conseguimos de la forma:

GUARDA "nombre de fichero" "nombre de procedimiento"

Para guardar un solo procedimiento.

GUARDA "nombre de fichero" [lista de procedimientos]

Para guardar más de uno.

Define tu propio juego de caracteres

Con los siguientes procedimientos creamos nuestras propias letras del abecedario. Son letras más grandes que las que aparecen en la pantalla y tienen una franja más gruesa en su lado izquierdo. Las hemos definido en mayúsculas.

Antes de nada definimos una serie de procedimientos que van a ser comunes en todas ellas.

```
? PARA AA
> BL
> REPITE 3 [AV 10 RE 10 GD 90
              AV 1 GI 90]
> FIN
```

Este procedimiento nos dibuja la franja ancha de las letras.

Los tres que vienen a continuación son simples repites de 2, 3 y 4 veces el dibujo de una línea de 10 puntos, girando a continuación 90 grados a la derecha. Como este proceso se repite varias veces en los procedimientos de las letras, lo más conveniente es hacerlos aparte para no tener que ponerlo cada vez que nos sea necesario.

```
? PARA R2
> REPITE 2 [AV 10 GD 90]
> FIN

? PARA R3
> REPITE 3 [AV 10 GD 90]
> FIN

? PARA R4
> REPITE 4 [AV 10 GD 90]
> FIN
```

Una vez que tenemos estos procedimientos definidos, podemos comenzar a definir las nuevas letras. Demos a cada procedimiento el nombre de la letra que dibuja.

```
? PARA A
> AA
> R3
> GD 90 AV 5 GI 90 AV 10
> SL RE 15 GD 90 RE 5
> FIN
```

Explicamos este procedimiento para que nos sirva de guía para los restantes.

Consulta los nombres de todos los procedimientos que se encuentran en la memoria.

EXPERIENCIA Y PRACTICAS EN LOGO

Introduce las siguientes órdenes:

? PM
? BP MT
? A

Habrás obtenido:



Fig. 3.

Antes de empezar a dibujar nuestras letras, la Tortuga tiene que estar orientada siempre hacia el norte. Lo primero que hace es dibujar la franja ancha (AA) y a continuación las tres líneas que forman la A(R3). Luego pintamos la línea que atraviesa y, por último, colocamos a la Tortuga en la posición adecuada para que comience a dibujar otra letra.

Este último proceso lo hacemos en todas las letras y consiste en situar a la Tortuga a cinco puntos de la letra dibujada y en la misma posición vertical y orientación con la que comenzó.

? PARA B

> AA
> R4
> AV 5 GD 90 AV 10
> SL AV 5 GI 90 RE 5
> FIN

? PARA C

> AA
> GI 90,RE 10
> R3
> SL AV 10 GI 90 AV 5 GI 90
> FIN

? PARA D

> AA
> AV 10 GD 90 AV 8
> GD 45 AV 2 GD 45
> AV 7 GD 45 AV 2
> GD 45 AV 8
> SL RE 15 GD 90
> FIN

? PARA E

> AA

> GI 90 RE 10
> R3
> SL AV 5 GD 90
> BL AV 10
> SL RE 15 GD 90 RE 5
> FIN

? PARA F

> AA
> R2
> SL AV 5 GD 90
> BL AV 10
> SL RE 15 GD 90 RE 5
> FIN

? PARA G

> AA
> GI 90 RE 10
> R3
> SL AV 5 BL
> GD 90 AV 3 RE 3
> GD 90 RE 5
> SL GD 90 AV 5 GI 90
> FIN

? PARA H

> AA
> AV 10 RE 5 GD 90
> AV 10 GI 90 AV 5
> RE 10
> SL GD 90 AV 5 GI 90
> FIN

? PARA I

> GD 90 AV 5 GI 90
> AA
> GI 90 AV 8 RE 13
> AV 5 GD 90 AV 10
> GD 90 AV 5 RE 13
> SL AV 18 GI 90 RE 10
> FIN

? PARA J

> BL AV 4 RE 4
> GD 90 AV 10 GI 90
> AA
> AV 10 GI 90 AV 8 RE 9
> SL RE 4 GD 90 RE 10
> FIN

? PARA K

> AA
> AV 10 RE 7 GD 44
> AV 10 SL RE 6
> GD 90 BL AV 8
> SL GI 44 AV 5 GI 90
> FIN

Puedes listar el contenido del área de trabajo.

? PARA L

> AA
> GI 90 RE 10
> R2
> SL AV 15 GI 90 RE 10
> FIN

? PARA M

> AA
> AV 10 GD 135 AV 8
> GI 90 AV 8 GD 135 AV 10
> SL GI 90 AV 5 GI 90
> FIN

? PARA N

> AA
> AV 10 GD 135 AV 14
> GI 135 AV 10
> SL GD 90 AV 5
> GI 90 RE 10

Por si tu ordenador no posee la letra Ñ,
llama "NN" al procedimiento que la dibuja.

> ? PARA NN

> N
> SL AV 12 GI 90
> AV 7 BL AV 8
> SL RE 15 GD 90 RE 12
> FIN

Fíjate que hemos utilizado dentro de
este procedimiento al que dibuja la N, y lo
único que le añadimos es la rayita de la N.

El procedimiento que dibuja la O lo llama-
mos O1, ya que la O no puede utilizarse por
ser una primitiva.

? PARA O1

> AA
> R4
> SL GD 90 AV 15 GI 90
> FIN

? PARA P

> AA
> R2
> AV 5 GD 90 AV 10
> SL RE 15 GD 90 RE 5
> FIN

? PARA Q

> AA
> R4
> GI 90 RE 10 GD 45
> AV 4 RE 6

> SL GD 135 AV 4 GI 90 AV 1
> FIN

? PARAR R

> AA
> R2
> AV 5 GD 90 AV 10
> GI 150 AV 11
> SL GI 30 AV 5 GI 90
> FIN

? PARA S

> SL AV 5 BL
> REPITE 2 [REPITE 3 [AV 5 RE 5 GD
90 AV 1 61 90] GD 90 AV 10 GD 90]
> SL RE 5 GD 90 RE 1
> BL AV 14 GI 90 SL
> AV 10 GI 90 BL AV 11
> SL RE 18 GD 90 RE 10
> FIN

? PARA T

> SL GD 90 AV 5 GI 90
> AA
> AV 10 GD 90 RE 8 AV 14
> SL AV 5 GI 90 RE 10
> FIN

? PARA U

> AA
> AV 10 RE 10 GD 90
> AV 10 GI 90 AV 10
> SL GD 90 AV 5 GI 90 RE 10
> FIN

? PARA V

> SL RE 1 GD 90 AV 5
> GI 120
> REPITE 4 [BL AV 13 SL RE 13
GD 120 AV 1 GI 120]
> BL GD 60 AV 13
> SL GD 60 AV 5 GI 90 RE 10
> FIN

? PARA X

> SL RE 1 GD 90 AV 10
> GI 135
> REPITE 4 [BL AV 15 SL RE 15 GD 135
AV 1 GI 135]
> GD 40 AV 12 GD 55
> BL RE 17 GD 40
> SL AV 18 GI 90
> FIN

? PARA Y1

> SL GD 90 AV 5

Existen teclas que te ahorran trabajo y tiempo.

EXPERIENCIA Y PRACTICAS EN LOGO

> GI 90 BL
> REPITE 3 [AV 4 RE 4 GD 90 AV 1 GI 90]
> AV 4 GD 45 AV 8 RE 8
> GI 135 AV 3 GD 45 AV 8
> SL GD 135 AV 20 GI 90 RE 10
> FIN

En este caso tampoco podemos usar como nombre Y para definir un procedimiento, ya que es una primitiva, por lo que utilizamos Y1.

? PARA Z
> BL GD 90 AV 13
> RE 13 GI 45
> REPITE 3 [AV 14 RE 14 GD 45 AV 1
GI 45]
> AV 14 GI 135 AV 13
> SL RE 18 GD 90 RE 10
> FIN

Definimos un procedimiento para poder dejar un espacio del tamaño de una letra entre palabras.

? PARA SP
> SL GD 90
> AV 10 GI 90
> FIN

Una vez que tenemos todas las letras definidas, podemos escribir en la pantalla lo que queramos. Sólo tendremos que situar a la Tortuga en el sitio donde queremos que empiece a escribir.

Las letras tienen una anchura de 10 puntos y la separación entre ellas es de 5 puntos; por tanto, si situas la Tortuga en el extremo izquierdo de tu pantalla, no escribas más de 14 letras, ya que no cabrían.

Si introduces las siguientes órdenes:

? PM OT
? BP SL
? PONPOS [-110 20]
? A B C D E F G
? H I J K L M
? SL PONPOS [-110 -10]
? N NN 01 P Q R S T U V X Y1 Z

Te aparecerán en la pantalla todas las letras.

Si quieres escribir en tu pantalla "HOLA, SOY LA TORTUGA DEL LOGO", deberás dar las siguientes órdenes:

? PM BP
? OT

? SL PONPOS [-100 70]
? H 01 L A SP S 01 Y1 SP L A
? SL PONPOS [-100 50]
? T 01 R T U G A S P D E L
? SL PONPOS [-50 30]
? L 01 G 01
? SL CENTRO MT

Gestión de la memoria de trabajo

Cuando tienes varios procedimientos definidos en la memoria y no recuerdas cuáles ni cuántos, puedes consultarle al LOGO.

? IMTS

Abreviatura de IMprime TítuloS.

En la pantalla aparecerán los nombres de todos los procedimientos que en ese momento halla en el espacio de trabajo.

Supongamos que tenemos definidos tres procedimientos (CUADRADO, CIRCULO y DIBUJO).

? IMTS

verás en la pantalla:

PARA CUADRADO
PARA CIRCULO
PARA DIBUJO

En algunas versiones podemos obtener la lista de títulos de los procedimientos contenidos en la memoria de la siguiente forma:

? ESCRIBE LPROCS

aparece en la pantalla:

CUADRADO CIRCULO DIBUJO

LPROCS significa lista de procedimientos. Puede usarse como dato de otras primitivas.

En la versión LOGO del SPECTRUM la orden que nos muestra los nombres de los procedimientos es ESCTS.

? ESCTS

Podemos visualizar el contenido de un procedimiento con la orden IM "nombre".

? IM "CUADRADO

y el de varios, detallando el nombre de cada uno de ellos:

? IM [CUADRADO CIRCULO DIBUJO]

Consulta el contenido del disquete.

En el primer caso sólo aparecerán las órdenes que componen el procedimiento CUADRADO; en el segundo aparecerán las órdenes que contiene cada uno de los procedimientos especificados.

Si sólo tenemos estos tres procedimientos en memoria, conseguimos lo mismo que en el último ejemplo con la orden:

? IM LPROCS

Donde LPROCS (lista de procedimientos) engloba a todos los procedimientos que tengamos definidos.

También puede ser que tu LOGO posea la orden IMPS.

? IMPS

Si la orden existe, en la pantalla te habrá aparecido lo mismo que con

? IM LPROCS

En el SPECTRUM la orden que lista un procedimiento determinado es:

ESCPROC "nombre"

? ESPROC "CUADRADO"

Y la que lista el contenido de todos los que tengamos en la memoria es:

? ESCPROCS

Otra cosa que puede sernos muy útil es saber cómo podemos borrar procedimientos del espacio de trabajo, bien porque ya no los necesitamos o bien para conseguir liberar parte de la memoria.

BORRA (nombre) o [lista de procedimientos]

Podemos borrar uno o varios a la vez.

? BORRA "CUADRADO"

Eliminas el procedimiento CUADRADO de la memoria, y si no lo tenías grabado en el disco, tendrás que volver a teclearlo otra vez cuando lo necesites.

? BORRA [CIRCULO DIBUJO]

Eliminas ambos procedimientos.

Si ahora tecleas:

? IMTS

verás que han desaparecido los nombres de los procedimientos que has borrado.

Si te interesa limpiar la memoria, quieres borrar todos los procedimientos, pon como parámetro de BORRA la lista completa de procedimientos.

? BORRA LPROCS

La orden que borra todos los procedimientos en el LOGO del SPECTRUM es:

? SUPPROCS

abreviación de SUPrime PROCedimientos.

Y para borrar uno o varios:

SUPRIME "nombre"

SUPRIME [lista de procedimientos]

Antes de borrar un procedimiento que puedes necesitar posteriormente asegúrate de que ya lo tienes guardado en el disco o en la cinta.

■ Las teclas y el editor I

Existe un número determinado de teclas en los ordenadores que tienen funciones específicas que realizar, como es el caso, por ejemplo, de la tecla DELETE o de la tecla STOP.

Ya hemos visto algunas de ellas, pero existe todavía un gran número que desconocemos y que su utilización nos va a permitir trabajar más rápido.

Vamos a conocer las que actúan en el editor del Logo, pudiendo también alguna de ellas hacerlo fuera.

Por lo general, en todos los ordenadores existen cuatro teclas que realizan la misma función. Se trata de las teclas de movimiento del cursor. Su representación y funciones son:

←: Desplaza el cursor un carácter a la izquierda.

↓: Desplaza el cursor a la línea inferior.

→: Desplaza el cursor un carácter a la derecha.

↑: Desplaza el cursor a la línea superior.

En el caso del SPECTRUM, estas funciones se obtienen pulsando respectivamente las teclas:

CAPS 5

CAPS 6

Quando el disquete esté lleno, recibirás un mensaje de aviso.

EXPERIENCIA Y PRACTICAS EN LOGO

CAPS 7

CAPS 8

En este ordenador, si pulsamos CAPS 0, se borra el carácter situado a la izquierda del cursor.

Dependiendo del ordenador, tenemos:

ORDENADORES MSX

CTRL A: Sitúa el cursor al inicio de la línea.

CTRL E: Sitúa el cursor al final de la línea.

CTRL G: Borra el carácter que está situado bajo el cursor.

Equivale a la tecla DEL (DELETE).

CTRL K: Borra toda la línea a partir del carácter que está situado bajo el cursor.

CTRL F: Desplaza al cursor un carácter hacia la derecha.

Equivale a la tecla de movimiento del cursor (→).

CTRL B: Desplaza al cursor un carácter hacia la izquierda.

Equivale a la tecla de movimiento del cursor (←).

Las teclas que hemos visto hasta ahora actúan tanto dentro del editor como fuera.

Algunas de las que sólo actúan en el editor son:

F1: Sale del editor tomando en cuenta las modificaciones realizadas en los procedimientos.

F5: Sale del editor, pero sin tomar en cuenta las modificaciones.

ORDENADOR SPECTRUM

E MODE CAPS 5: Sitúa el cursor al principio de la línea.

E MODE CAPS 8: Sitúa el cursor al final de la línea.

E MODE Y: Borra la línea a partir de la posición en donde se encuentra el cursor.

CAPS BREAK/SPACE: Sale del editor sin tomar en cuenta las modificaciones.

E MODE C: Sale del editor tomando en cuenta las modificaciones.

ORDENADORES PC-COMPATIBLES

F9: Desplaza al cursor hasta el principio de la línea en donde se encuentra.

F10: Desplaza al cursor hasta el final de la línea.

F7: Borra la línea desde la posición en donde se encuentra el cursor.

F8: Inserta una línea a partir de la posición en la que está el cursor. (Sólo cuando estamos dentro del editor.)

ALT F2: Sale del editor sin tomar en cuenta las modificaciones realizadas.

F1: Sale del editor tomando en cuenta las modificaciones.

Estas mismas funciones las realizan otras versiones del Logo para estos ordenadores, pero utilizando las siguientes teclas:

⇐ : Para situar al cursor al final de la línea.

SHIFT y ⇐ : Para situar al cursor al principio de la línea.

CTRL y → : Borra la línea a partir de la posición en donde está el cursor.

INS: Inserta una línea a partir de donde se encuentra el cursor. (Sólo dentro del editor.)

F10: Sale del editor sin tomar en cuenta las modificaciones.

F1 Sale del editor tomando en cuenta las modificaciones.

■ Mensajes Logo

A medida que avanzamos con el Logo y vamos descubriendo y aprendiendo cosas nuevas, nos van apareciendo diferentes mensajes que nos da el Logo. Es necesario saber su significado, para así poder realizar aquello que nos comunica o bien corregir el error que hayamos cometido.

Ya vimos algunos mensajes y explicamos su significado.

Veamos algunos mensajes más que nos pueden aparecer cuando estamos trabajando.

— Nombre de procedimiento DEFINIDO.

Hemos definido un procedimiento y todas las órdenes que hemos dado han sido tomadas en cuenta. Este mensaje aparece tanto cuando terminamos de definir un procedimiento fuera del editor como cuando salimos del editor y en él hemos estado trabajando con un procedimiento.

— Nombre de procedimiento YA ESTA DEFINIDO.

Al guardar no utilices nombres de fichero que ya existan.

Estamos utilizando un nombre para definir un procedimiento, que ya hemos utilizado anteriormente para definir otro.

— NO SE COMO HACER PARA nombre

El nombre que se ha especificado no corresponde a un título de procedimiento.

— NO ACEPTA nombre COMO DATO.

Se ha dado un dato que no es válido en una primitiva o en un procedimiento.

— n PROCEDIMIENTO(S) GUARDADO(S)

Han sido guardados en algún soporte externo el número de procedimientos que determina n. (Este mensaje no aparece en algunas versiones.)

— Nombre de archivo YA EXISTE.

El nombre que damos al archivo ya ha sido utilizado anteriormente y pertenece a otro archivo que se encuentra en el disco con el que estamos trabajando.

— FALTAN ELEMENTOS EN

Los datos que se especifican no son suficientes.

■ Cuadro resumen

— ESCRIBE ESCALA

Muestra en pantalla uno o dos números que determinan la escala actual de la pantalla gráfica.

— PONESCALA [x y] o PONESCALA n

Asigna a la pantalla una nueva escala.

— IMTS

Muestra en pantalla los nombres de todos los procedimientos existentes en el área de trabajo.

En el SPECTRUM:

ESCTS

— IM "nombre/[lista]

Lista el contenido del o de los procedimientos especificados.

En el SPECTRUM:

ESCPROC "nombre/[lista]

— IMPS o IM "LPROCS

Lista el contenido de todos los procedimientos que estén en memoria.

En el SPECTRUM:

ESCPROCS

— BORRA "nombre/[lista]

Borra del espacio de trabajo el o los procedimientos especificados.

En el SPECTRUM:

SUPRIME "nombre/[lista]

— BORRA LPROCS

Borra de la memoria todos los procedimientos.

En el SPECTRUM:

SUPPROCS

— DIR

Orden que visualiza los nombres de todos los ficheros contenidos en el disco (por defecto) y el espacio disponible.

En el SPECTRUM:

CATALOGO

■ Ejercicios

1. Define una serie de procedimientos para dibujar una regla milimetrada.

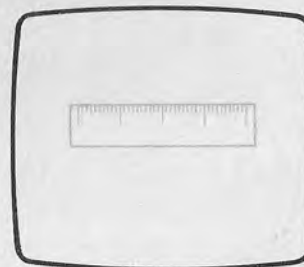


Fig. 4.

2. Define un procedimiento que muestre la lista de los nombres de los procedimientos que tengas en el área de trabajo y los lleve a todos al editor.
¿Sabrías decir si entre estos procedimientos aparecerá el que tú acabas de definir?
3. Define un procedimiento que llame a uno que dibuje una tubería y a otro que dibuje una cubeta, como en la figura.

Borra todo lo que no te haga falta.

EXPERIENCIA Y PRACTICAS EN LOGO

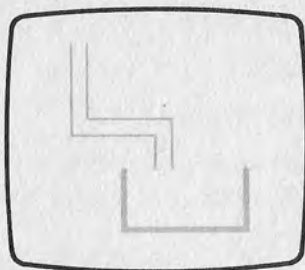


Fig. 5.

Una vez que hayas realizado esto, haz un procedimiento que llene la cubeta de agua.

4. Intenta dibujar la última figura del ajedrez que nos falta: EL CABALLO.

5. ¿Son correctas las siguientes órdenes?:

- PONESCALA 100 20
- IM "AVANZA
- IM "CASA
- SUPRIME [CIRCULO CUADRADO CASA VENTANA]
- IMTS LPROCS
- ED LPROCS

■ Solución de los ejercicios

1: Procedimiento para dibujar la forma de la regla.

```
? PARA RECTAN
> REPITE 2 [AV 20 GD 90 AV 160 GD 90]
> FIN
```

Procedimiento para milimetrar la regla.

```
? PARA MILIM
> REPITE 4 [REPITE 9 [GI 90 AV 4 GD
90 AV 3 RE 3] GI 90 AV 4 GD 90 AV 10 RE 10]
> FIN
```

Procedimiento que inicializa la pantalla, posiciona a la Tortuga y llama a los dos anteriores.

```
? PARA REGLA
> BP
> SL
> PONX -80
> BL
> RECTAN
> AV 20
> GD 180
> MILIM
> FIN
```

Para dibujar la regla completa teclea:

? REGLA

2:

```
> PARA LISTARP
> BP
> IMTS
> EDITA LPROS
> FIN
```

Para que se ejecute teclea:

? LISTARP

Este procedimiento aparecerá en la lista de nombres y en el editor, ya que si podemos ejecutarlo es porque lo tenemos definido dentro de la memoria.

3:

Procedimiento de la tubería:

```
? PARA TUBERIA
> PONCL 1
> SL
> PONPOS [-100 50]
> BL
> REPITE 2 [RE 50 GI 90]
> AV 8 SL
> PONPOS [-97 50]
> PONRUMBO 0
> BL
> REPITE 2 [RE 47 GI 90]
> GI 90 AV 3
> GD 90 AV 11
> FIN
```

Para dibujar la cubeta hacemos el siguiente procedimiento:

```
? PARA CUBETA
> PONCL 14
> SL
> PONPOS [-58 -10]
> PONRUMBO 0
> BL RE 35 GD 90
> AV 67 GI 90
> AV 35 SL
> PONPOS [-57 -10]
> BL RE 34 GD 90
> AV 65 GI 90 AV 34
> FIN
```

Ahora hacemos uno que contenga a los dos.

Antes de empezar a teclear, estructura el ejercicio.

? PARA TUBYCUB

> PM
> BP
> OT
> TUBERIA
> CUBETA
> FIN

El procedimiento que lleva la cubeta de agua es:

? PARA AGUA
> PONCL 5
> SL
> PONPOS [-48 8]
> BL
> REPITE 35 [RE 1 ESPERA 0.5]
> GD 90 RE 8
> REPITE 25 [AV 63 RE 63 GI 90 AV 1
GD 90]
> AV 63 RE 55 GI 90
> GOMA AV 9
> FIN

Si ahora introduces:

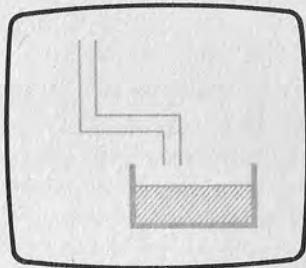
TUBYCUB

Obtendremos el dibujo de la tubería y de la cubeta.

Para llenarla de agua:

? AGUA

Tendrá que aparecer ahora la cubeta de la siguiente forma:



4:

Esta puede ser la forma de dibujar un caballo de ajedrez de una forma sencilla:

? PARA CABALLO

INICIALIZACION

> PM
> SL

> BP

> OT

CENTRANDO DIBUJO

> GI 90 AV 35 GD 90
> RE 6 BL

DIBUJO

> REPITE 2 [AV 10 GD 90 AV 70 GD 90]
> GI 90 RE 5 GD 90
> SL AV 10 BL
> AV 5 GD 90 AV 60
> GD 90 AV 5 RE 5
> GD 90 AV 15 GD 90
> AV 10 GI 90 AV 28
> RE 28 GD 150
> AV 10 GI 60 AV 1
> GI 90 AV 47 RE 47
> GD 90 AV 5 GI 90
> AV 47 RE 47 GD 90
> REPITE 8 [AV 10 GI 4]
> GI 60
> REPITE [5 AV 10 GI 15]
> GI 60 AV 8 GI 45
> REPITE 3 [AV 10 GI 20]
> REPITE 6 [RE 10 GI 3]
> GD 170 AV 6 GI 60
> AV 12 GD 60 AV 10
> FIN

5:

- PONESCALA 100 20: INCORRECTA. Faltan los corchetes.
- IM "AVANZA: INCORRECTA. AVANZA es una primitiva del LOGO y no podemos utilizarla como nombre de procedimiento.
- IM "CASA: INCORRECTA siempre que el procedimiento CASA se encuentre en el espacio de trabajo.
- SUPRIME [CIRCULO CUADRADO CASA VENTANA]: CORRECTA si los procedimientos especificados están definidos en la memoria.
- IMTS LPROCS: INCORRECTA. IMTS no necesita ningún parámetro.
- ED LPROCS: CORRECTA. Todos los procedimientos que se encuentran en el área de trabajo pasarán al editor.

Ten cuidado al borrar, no vayas a suprimir algo útil.

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

Movimiento diagonal de figuras complejas



NA vez que hemos visto cómo hacer que un dibujo compuesto de más de un carácter se mueva por la pantalla horizontal y verticalmente, podemos meternos en el estudio del movimiento diagonal de figuras complejas.

Como siempre, y antes de entrar en materia, os recomendamos que introduzcáis el programa 1 para ver en qué consiste este tipo de movimiento.

```
10 REM *****
20 REM * MOVIMIENTO DE UN COCHE POR LA *
30 REM * CARRETERA CON MOVIMIENTO *
40 REM * POSITIVO-POSITIVO *
50 REM *****
60 REM
70 REM *** DIBUJO DE LA CARRETERA ***
80 REM
90 CLS
100 FOR I=1 TO 16
110 LOCATE I+4,I
120 PRINT "*"
130 LOCATE I+4,I+7
140 PRINT "*"
150 NEXT I
160 REM
170 REM *** DEFINICION DEL COCHE ***
180 REM
190 LET A$="O\\O"
200 LET B$=" \ \"
210 REM
220 REM *** MOVIMIENTO DEL COCHE ***
```

```
230 REM
240 FOR I=3 TO 17
250 LOCATE I,I-2
260 PRINT " "
270 LOCATE I+1,I-1
280 PRINT A$
290 LOCATE I+2,I
300 PRINT B$
310 LOCATE I+3,I+1
320 PRINT A$
330 FOR J=1 TO 50
340 NEXT J
350 NEXT I
360 END
```

Programa 1.

Antes de ejecutar el programa, y si tu ordenador no es ni un IBM ni un AMSTRAD, mira las modificaciones que tienes que realizar para que el programa pueda funcionar. Estas modificaciones son:

COMMODORE:

```
90 PRINT "<SHIFT-HOME>"
110 POKE 214,I+4:POKE 211,I
130 POKE 214,I+4:POKE 211,I+7
250 POKE 214,I:POKE 211,I-2
270 POKE 214,I+1:POKE 211,I-1
290 POKE 214,I+2:POKE 211,I
310 POKE 214,I+3:POKE 211,I+1
```

MSX:

```
110 LOCATE I,I+4
130 LOCATE I+7,I+4
250 LOCATE I-2,I
270 LOCATE I-1,I+1
290 LOCATE I,I+2
310 LOCATE I+1,I+3
```

SPECTRUM

```
110 PRINT AT I+4,I;  
130 PRINT AT I+4,I+7;  
250 PRINT AT I,I-2;  
270 PRINT AT I+1,I-1;  
290 PRINT AT I+2,I;  
310 PRINT AT I+3,I+1;  
360 STOP
```

El funcionamiento del programa, como todo lo que hemos visto hasta ahora, es muy sencillo. Veamos su funcionamiento línea a línea.

Línea 90. Borrarnos la pantalla. No debemos olvidar que en el COMMODORE, para borrarla, tenemos que poner un corazón entre las comillas de un PRINT. Dicho corazón se consigue pulsando la tecla SHIFT y, sin soltarla, pulsar la tecla HOME. Este corazón le dice al COMMODORE que tiene que borrar la pantalla.

Línea 100. Comienza un bucle. Este bucle se encarga de dibujar la carretera por la que se moverá el coche. Dicha carretera no es necesaria, pero nos ayuda a ver que el camino que sigue el coche es una diagonal de 45 grados con respecto a la horizontal.

Línea 110. Se coloca el cursor en una cierta posición para imprimir el borde izquierdo de la carretera. Dicha posición va variando con el bucle, con lo que conseguimos que cada vez se coloque el cursor una línea más abajo y una línea más a la derecha.

Línea 120. Se imprime un asterisco (*) que hará las veces de borde de la carretera.

Línea 130. El cursor se coloca ahora en el borde derecho de la carretera. También va variando la posición de este borde hacia abajo y hacia la derecha para ir paralelo al borde izquierdo.

Línea 140. Se imprime otro asterisco. Este hará las veces del borde derecho de la carretera.

Línea 150. Aquí termina el bucle que se encarga de dibujar la carretera.

Línea 190. En esta línea, y en la siguiente, se define el coche tal y como aparecerá en la pantalla. El coche que utilizamos puede verse en la figura 1. Como puedes apreciar, la parte de delante y la parte de atrás del coche son iguales. Por ello, sólo definimos una sola variable alfanumérica (A\$).

Línea 200. En esta línea se define el resto del coche.

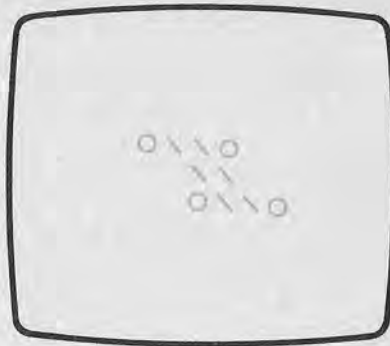


Fig. 1. Para dibujar el coche utilizamos el SLASH y la 'O' mayúscula.

Línea 240. Comienza otro bucle cuya función es realizar el movimiento del coche. Este bucle empieza con el valor tres y termina con el valor 17. Esto es así porque la carretera empieza en la línea 4 y termina en la línea 20. Como el coche ocupa tres filas de alto, cuando estemos localizados en la línea 3, el coche llegará hasta la 5, la cual es parte de la carretera. El bucle termina en la línea 17, porque cuando el coche llegue a esta línea su morro llegará hasta la línea 20, que es donde termina la carretera. Aquellas personas que tengan un ordenador con más de 21 líneas por pantalla, pueden variar la longitud del bucle a 19 ó 20 para que el coche recorra más espacio.

Línea 250. Se coloca el cursor en medio de las dos filas de asteriscos.

Línea 260. Se imprimen cuatro blancos. En la primera vuelta del bucle esto no sirve para nada, pero a partir de la segunda, y hasta la última, estos cuatro blancos sirven para borrar la parte trasera del coche que había anteriormente dibujado. Este borrado, conjugado con la impresión del coche una línea más abajo y un carácter más a la derecha, nos dará la sensación de movimiento que estábamos buscando.

Línea 270. Se coloca el cursor una línea más abajo (recuerda que el coche está formado por 3 líneas de 4 caracteres) y un carácter más a la derecha de la posición anterior.

Línea 280. Se imprime la parte delantera del coche, o lo que es igual, la variable alfanumérica A\$.

Línea 290. Se posiciona de nuevo el cursor una línea más abajo y un carácter más a la izquierda que donde se encontraba anteriormente. Como puedes apreciar, la forma del coche, al igual que la de la carretera, es diagonal u oblicua. Si no tuviese el coche esta forma, el movimiento quedaría muy desfigura-

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

do y realmente no sería un movimiento diagonal.

Línea 300. Imprimimos la parte central del coche que se encuentra almacenada en la variable B\$.

Línea 310. Volvemos a colocar el cursor una línea más abajo y un carácter más a la derecha para imprimir la siguiente línea.

Línea 320. Imprimimos la parte delantera del coche (que es igual a la parte trasera).

Línea 330. Aquí comienza un pequeño bucle cuya función es, durante un rato, no hacer absolutamente nada. Con este bucle se consigue que el coche no vaya muy de prisa. Variando el número que se encuentra después de la palabra **TO**, se puede hacer que el coche se mueva más despacio o más de prisa.

Línea 340. Aquí termina el bucle de retardo.

Línea 350. Se acaba el bucle que rige el movimiento del coche.

Línea 360. Se acaba el programa.

En las variaciones que se proponen para que el programa funcione en el COMMODORE, se han utilizado dos POKES que antes no conocíamos. Estos son el 214 y el 211. La función de estos POKES es la de colocar el cursor en la posición de la pantalla que nosotros le POKEEMOS. Con este nuevo truco nos ahorramos el tener que utilizar la rutina LOCATE PARA COMMODORE que se dio en el primer tomo. La forma de usar estos dos POKES es la siguiente:

1. En la dirección 214 podemos introducir un número entre 0 y 24. Este número le dice al ordenador la fila en la que queremos que se coloque el cursor.

2. En la dirección 211 podemos introducir un número entre 0 y 39. Este número le dice al ordenador la columna a la que nos queremos mover con el cursor.

Por ejemplo, si queremos imprimir la palabra COMMODORE en la columna 13 de la fila 7, escribiremos la siguiente línea:

```
POKE 214,7:POKE 211,13:PRINT "COMMODORE"
```

con lo que dicha palabra nos aparecerá donde nosotros hemos elegido, tal y como puede verse en la figura 2.

Como vimos cuando hablamos del movimiento diagonal de figuras compuestas de un solo carácter, había cuatro direcciones dis-



Fig. 2. Usando los POKES 214 y 211 podemos colocar, en el COMMODORE, el cursor donde queramos.

tintas para este tipo de movimiento. Estas eran:

1. Positivo-Positivo. El valor de la coordenada X y el de la coordenada Y van aumentando. No hace falta que ambos aumenten a la vez. Puede aumentar la X de uno en uno y la Y de dos en dos.

2. Positivo-Negativo. El valor de la coordenada X va aumentando, mientras el de la coordenada Y va disminuyendo. Como en el caso anterior, el incremento (o decremento) de las dos variables no ha de ser igual.

3. Negativo-Negativo. El valor de la coordenada X va disminuyendo, mientras el de la coordenada Y va aumentando. El incremento puede ir de cualquier manera.

4. Negativo-Positivo. En éste, el último caso posible, ambas variables van disminuyendo, no teniendo que ser proposicional el decremento de ambas.

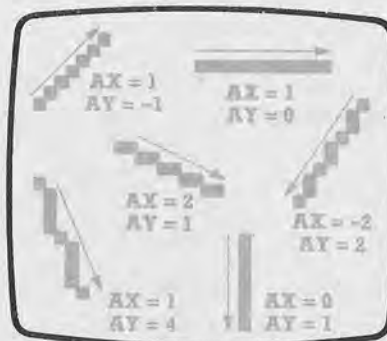


Fig. 3. La relación entre la variación de la componente X y la componente Y en un movimiento no tiene por qué ser igual a 1.

Una vez que hemos repasado estos conceptos, vamos a ver una ampliación del programa 1. En este programa (el 2) veremos cómo el movimiento pasa de ser Positivo-Positivo a ser Positivo-Negativo al llegar el coche a la última línea de la pantalla.

```

10 REM *****
20 REM * MOVIMIENTO DIAGONAL DE UN COCHE *
30 REM * POSITIVO-POSITIVO Y POSITIVO- *
40 REM * NEGATIVO POR UNA CARRETERA *
50 REM *****
60 REM
70 REM *** DIBUJO DE LA CARRETERA ***
80 CLS
90 REM
110 REM
120 REM <<< BORDE SUPERIOR >>>
130 REM
140 FOR I=1 TO 12
150   LOCATE I,I+3
160   PRINT "*"
170   LOCATE I,28-I
180   PRINT "*"
190 NEXT I
200 REM
210 REM <<< BORDE INFERIOR >>>
220 REM
230 FOR I=1 TO 15
240   LOCATE I+4,I
250   PRINT "*"
260   LOCATE I+4,31-I
270   PRINT "*"
280 NEXT I
290 REM
300 REM *** DEFINICION DEL COCHE ***
310 REM
320 REM <<< COCHE HACIA ABAJO >>>
330 REM
340 LET A$="O\\O"
350 LET B$="  \\ "
360 REM
370 REM <<< COCHE HACIA ARRIBA >>>
380 REM
390 LET C$="O//O"
400 LET D$="  // "
410 REM
420 REM *** MOVIMIENTO DIAGONAL DEL COCHE ***
430 REM
440 REM <<< MOVIMIENTO POSITIVO-POSITIVO >>>
450 REM
460 FOR I=3 TO 13
470   LOCATE I,I-2
480   PRINT "  "
490   LOCATE I+1,I-1
500   PRINT A$
510   LOCATE I+2,I
520   PRINT B$
530   LOCATE I+3,I+1
540   PRINT A$
550   FOR J=1 TO 50
560     NEXT J
570 NEXT I
571 REM
572 REM *** BORRAMOS EL COCHE ***
573 REM

```

```

574 FOR I=14 TO 16
575   LOCATE I,I-2
576   PRINT "  "
577 NEXT I
580 REM
590 REM <<< MOVIMIENTO POSITIVO-NEGATIVO >>>
600 REM
610 FOR I=16 TO 4 STEP -1
620   LOCATE I,30-I
630   PRINT "  "
640   LOCATE I-1,31-I
650   PRINT C$
660   LOCATE I-2,32-I
670   PRINT D$
680   LOCATE I-3,33-I
690   PRINT C$
700   FOR J=1 TO 50
710     NEXT J
720 NEXT I
730 END

```

Programa 2.

Para que el programa pueda funcionar en ordenadores distintos del AMSTRAD e IBM (y compatibles) hay que realizar las siguientes modificaciones:

COMMODORE:

```

90 PRINT "<SHIFT-HOME>"
150 POKE 214,I:POKE 211,I+3
170 POKE 214,I:POKE 211,28-I
240 POKE 214,I+4:POKE 211,I
260 POKE 214,I+4:POKE 211,31-I
470 POKE 214,I:POKE 211,I-2
490 POKE 214,I+1:POKE 211,I-1
510 POKE 214,I+2:POKE 211,I
530 POKE 214,I+3:POKE 211,I+1
575 POKE 214,I:POKE 211,I-2
620 POKE 214,I:POKE 211,30-I
640 POKE 214,I-1:POKE 211,31-I
660 POKE 214,I-2:POKE 211,32-I
680 POKE 214,I-3:POKE 211,33-I

```

MSX

```

150 LOCATE I+3,I
170 LOCATE 28-I,I
240 LOCATE I,I+4
260 LOCATE 31-I,I+4
470 LOCATE I-2,I
490 LOCATE I-1,I+1
510 LOCATE I,I+2
530 LOCATE I+1,I+3
575 LOCATE I-2,I
620 LOCATE 30-I,I
640 LOCATE 31-I,I-1
660 LOCATE 32-I,I-2
680 LOCATE 33-I,I-3

```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

SPECTRUM

```
150 PRINT AT I,I+3;
170 PRINT AT I,28-I;
240 PRINT AT I+4,I;
260 PRINT AT I+4,31-I;
470 PRINT AT I,I-2;
490 PRINT AT I+1,I-1;
510 PRINT AT I+2,I;
530 PRINT AT I+3,I+1;
575 PRINT AT I,I-2;
620 PRINT AT I,30-I;
640 PRINT AT I-1,31-I;
660 PRINT AT I-2,32-I;
680 PRINT AT I-3,33-I;
730 STOP
```

Este programa es la unión del anterior con algunas nuevas instrucciones que hemos incluido para que se realice el movimiento diagonal Positivo-Negativo. También ha habido que variar la rutina que dibuja la carretera, y ha sido necesario borrar el coche cuando llega al final del movimiento Positivo-Positivo.

Analicemos línea a línea cómo funcionan las variaciones que hemos introducido en el programa 2.

Este primer grupo de líneas son las encargadas de dibujar la carretera.

Línea 140. Comienza un bucle que nos permitirá dibujar el borde superior de la carretera.

Línea 150. Colocamos el cursor en la parte izquierda de la pantalla para, en la línea siguiente, imprimir un asterisco. Según vaya avanzando el bucle, el cursor se irá colocando una línea más abajo y un carácter más a la derecha.

Línea 160. Imprimimos un asterisco que actuará como borde de la pantalla.

Línea 170. Se coloca el cursor en la parte derecha de la pantalla. Esta posición irá variando con el bucle de manera que, a cada vuelta de éste, el cursor se colocará una línea más abajo y un carácter más a la izquierda.

Línea 180. Se imprime un asterisco.

Línea 190. Aquí termina el bucle.

Línea 230. Ahora, y gracias a este bucle, vamos a dibujar el borde inferior de la carretera.

Línea 240. Colocamos el cursor a la izquierda de la pantalla. En cada vuelta del bucle, el cursor se colocará una línea más abajo y un carácter más a la derecha que la vez anterior.

Línea 250. Se imprime un asterisco. Actuará como límite de la carretera.

Línea 260. Realiza lo mismo que la 240, pero empezando por la derecha de la pantalla. La posición del cursor se irá desplazando hacia abajo y hacia la izquierda con cada vuelta del bucle de la línea 230.

Línea 270. Se termina el bucle encargado de dibujar el borde inferior de la carretera.

A continuación se define el coche tal como se hizo en el programa 1. Como en este programa el coche cambia de sentido, necesitamos una nueva definición del coche con una orientación distinta. Esto lo realizamos en las líneas 390 y 400.

Desde la línea 460 y hasta la 570 se realiza el movimiento Positivo-Positivo del coche. Esta parte es igual que la que vimos en el programa 1.

Una vez que termina el movimiento diagonal hacia abajo del coche, tenemos que borrarlo para seguir con el movimiento diagonal hacia arriba. Esto lo realizamos en las siguientes líneas:

Línea 574. Se empieza un bucle de tres pasos, desde la línea 14 a la 16, dentro del cual se borrará el coche.

Línea 575. Se coloca el cursor en la línea que especifica el bucle y en la columna I-2, pues es en esta posición donde se encuentra el coche. Con cada vuelta del bucle, aparte de bajar una línea, nos movemos un lugar hacia la derecha para ir de acuerdo con la forma escalonada del coche.

Línea 576. Se imprimen cuatro blancos cuya función es borrar la línea correspondiente del coche.

Línea 577. Se termina este bucle.

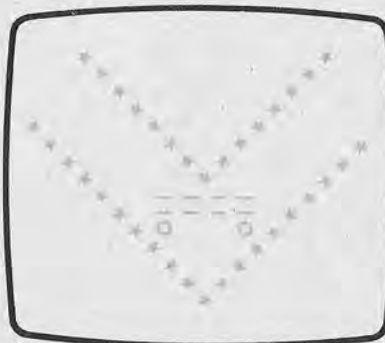


Fig. 4. ¿Podrías variar el programa 2 para que, al terminar el movimiento Positivo-Positivo dibujases el coche como aparece en el dibujo?

La última parte del programa es la encargada de realizar el movimiento Positivo-Negativo del coche.



Fig. 5. Dibujo del coche cuando realiza el movimiento Positivo-Negativo.

Línea 610. Aquí comienza el bucle. El incremento del bucle es negativo porque el movimiento va desde la línea 16 de la pantalla a la 4.

Línea 620. Colocamos el cursor dentro de la carretera. En la línea siguiente se imprimen cuatro espacios en blanco. En la primera vuelta del bucle esto no sirve para nada, pero, a partir de la segunda, tiene como función borrar la última línea que configura el coche.

Línea 630. Se imprimen cuatro espacios en blanco.

Línea 640. Se coloca el cursor una línea más arriba y un carácter más a la derecha que lo que acabamos de borrar.

Línea 650. Y se imprime la primera línea del coche.

Línea 660. Se coloca el cursor una línea más arriba y un carácter más a la derecha que en la línea 640.

Línea 670. Y se imprime la segunda línea del coche.

Línea 680. Se realiza la misma operación que en las líneas 64 y 660.

Línea 690. Y se imprime la última línea que configura el dibujo del coche y que corresponde con la primera.

Líneas 700 y 710. Se realiza un bucle de retardo de 50 pasos para hacer que el coche se mueva más lentamente.

Línea 720. Aquí termina el bucle principal que se encarga del movimiento Positivo-Negativo del coche.

Línea 730. Termina el programa.

Una vez entendido cómo funciona este último programa, podemos modificarlo para que parezca que las ruedas se están moviendo. Esto lo podemos conseguir sustituyendo,

de vez en cuando, las O's mayúsculas por o's minúsculas. Te aconsejo que, antes de introducir el programa 3, intentes resolver tú mismo el problema.

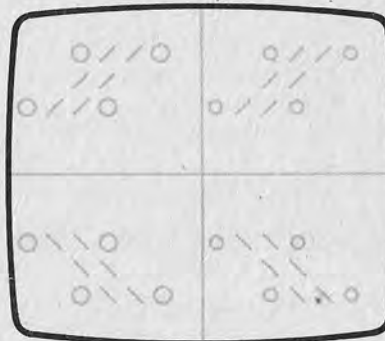


Fig. 6. Dibujo de los cuatro dibujos del coche que utilizamos en el programa 3.

```

10 REM *****
20 REM * MOVIMIENTO DIAGONAL DE UN COCHE *
30 REM * POSITIVO-POSITIVO Y POSITIVO- *
40 REM * NEGATIVO POR UNA CARRETERA CON *
50 REM * MOVIMIENTO DE LAS RUEDAS *
60 REM *****
70 REM
80 REM *** DIBUJO DE LA CARRETERA ***
90 CLS
100 REM
120 REM
130 REM <<< BORDE SUPERIOR >>>
140 REM
150 FOR I=1 TO 12
160   LOCATE I,I+3
170   PRINT "*"
180   LOCATE I,28-I
190   PRINT "*"
200 NEXT I
210 REM
220 REM <<< BORDE INFERIOR >>>
230 REM
240 FOR I=1 TO 15
250   LOCATE I+4,I
260   PRINT "*"
270   LOCATE I+4,31-I
280   PRINT "*"
290 NEXT I
300 REM
310 REM *** DEFINICION DEL COCHE ***
320 REM
330 REM <<< COCHE HACIA ABAJO >>>
340 REM
350 DIM A$(2)
360 LET A$(1)="O\O"
370 LET A$(2)="o\o"
380 LET B$="--"
390 REM
400 REM <<< COCHE HACIA ARRIBA >>>

```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

```

410 REM
420 DIM C$(2)
430 LET C$(1)="O//O"
440 LET C$(2)="O//O"
450 LET D$=" // "
460 REM
470 REM *** MOVIMIENTO DIAGONAL DEL COCHE ***
480 REM
490 REM <<< MOVIMIENTO POSITIVO-POSITIVO >>>
500 REM
510 FOR I=3 TO 12 STEP 2
520   FOR J=0 TO 1
530     LOCATE I+J,I+J-2
540     PRINT "   "
550     LOCATE I+J+1,I+J-1
560     PRINT A$(J+1)
570     LOCATE I+J+2,I+J
580     PRINT B$
590     LOCATE I+J+3,I+J+1
600     PRINT A$(J+1)
610     FOR K=1 TO 50
620       NEXT K
630   NEXT J
640 NEXT I
650 REM
660 REM *** BORRAMOS EL COCHE ***
670 REM
680 FOR I=13 TO 15
690   LOCATE I,I-2
700   PRINT "   "
710 NEXT I
720 REM
730 REM <<< MOVIMIENTO POSITIVO-NEGATIVO >>>
740 REM
750 FOR I=15 TO 5 STEP -2
760   FOR J=1 TO 0 STEP -1
770     LOCATE I+J,30-I-J
780     PRINT "   "
790     LOCATE I+J-1,31-I-J
800     PRINT C$(J+1)
810     LOCATE I+J-2,32-I-J
820     PRINT D$
830     LOCATE I+J-3,33-I-J
840     PRINT C$(J+1)
850     FOR K=1 TO 50
860       NEXT K
870   NEXT J
880 NEXT I
890 END

```

Programa 3.

Como siempre, y antes de ver el funcionamiento del programa, vamos a dar las modificaciones necesarias para todos los ordenadores distintos del AMSTRAD y del IBM pc.

COMMODORE:

```

90 PRINT "<SHIFT-HOME>"
160 POKE 214,I:POKE 211,I+3
180 POKE 214,I:POKE 211,28-I
250 POKE 214,I+4:POKE 211,I
270 POKE 214,I+4:POKE 211,31-I
530 POKE 214,I+J:POKE 211,I+J-2
550 POKE 214,I+J+1:POKE 211,I+J-1
570 POKE 214,I+J+2:POKE 211,I+J
590 POKE 214,I+J+3:POKE 211,I+J+1
690 POKE 214,I:POKE 211,I-2
770 POKE 214,I+J:POKE 211,30-I-J
790 POKE 214,I+J-1:POKE 211,31-I-J
810 POKE 214,I+J-2:POKE 211,32-I-J
830 POKE 214,I+J-3:POKE 211,33-I-J

```

MSX:

```

160 LOCATE I+3,I
180 LOCATE 28-I,I
250 LOCATE I,I+4
270 LOCATE 31-I,I+4
530 LOCATE I+J-2,I+J
550 LOCATE I+J-1,I+J+1
570 LOCATE I+J,I+J+2
590 LOCATE I+J+1,I+J+3
690 LOCATE I-2,I
770 LOCATE 30-I-J,I+J
790 LOCATE 31-I-J,I+J-1
810 LOCATE 32-I-J,I+J-2
830 LOCATE 33-I-J,I+J-3

```

SPECTRUM:

```

160 PRINT AT I,I+3;
180 PRINT AT I,28-I;
250 PRINT AT I+4,I;
270 PRINT AT I+4,31-I;
530 PRINT AT I+J,I+J-2;
550 PRINT AT I+J+1,I+J-1;
570 PRINT AT I+J+2,I+J;
590 PRINT AT I+J+3,I+J+1;
690 PRINT AT I,I-2;
770 PRINT AT I+J,30-I-J;
790 PRINT AT I+J-1,31-I-J;
810 PRINT AT I+J-2,32-I-J;
830 PRINT AT I+J-3,33-I-J;
890 STOP

```

El programa funciona igual que el 2, pero con la diferencia que, para realizar el movimiento, se utilizan dos bucles. Uno dentro del otro. El bucle más interno (el que tiene como variable índice a **J**) es el encargado de imprimir; bien el coche con las ruedas grandes, bien el que tiene las ruedas pequeñas. Como el bucle va variando entre 0 y 1, esto significa que cada vez se imprimirá un tipo distinto de coche. Como puede verse en el

programa, la definición de los coches es distinta de la que se da en el programa 2. En este programa se almacenan los distintos tipos de ruedas en la variable dimensionada A\$. Esta variable tiene sólo dos elementos, lo cual la hace idónea para utilizarla con el bucle interno.

Por otro lado, el bucle externo (el que utiliza la variable **I**) tiene un incremento de 2 (o de -2, según la dirección del movimiento); esto está hecho así para poder sumar el valor de **I** con el valor de **J**. Con ello se consigue que, con un cambio muy pequeño, no sólo se muevan las ruedas, sino que, además, el movimiento sea exactamente igual que el visto en el programa anterior.

Ya para terminar, os propongo un programa que veremos en el próximo tomo. Dicho programa tiene que mover un gusano por toda la pantalla con movimiento VERTICAL,

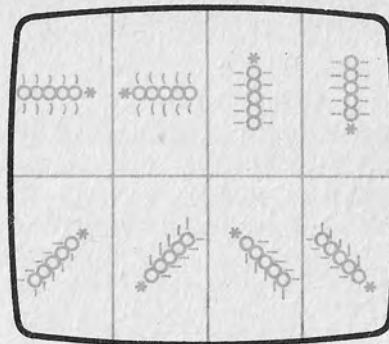
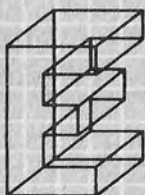


Fig. 7. Estas pueden ser las definiciones del gusano que aparece en el programa que se propone.

HORIZONTAL y DIAGONAL. Hay que contemplar los posibles choques del gusano con los bordes de la pantalla. Para que el programa sea más completo, se propone que la elección de la dirección y el sentido del movimiento sea aleatoria.

Trucos de programación



N los dos últimos tomos hemos visto cómo tendría que ser un programa de gestión elemental de ficheros, qué funciones tendría que realizar, qué rutinas serían necesarias y cómo organizaríamos los datos. Hasta el mo-

mento, y en el tomo anterior, hemos visto algunas de las rutinas que necesitaremos para realizar el programa. Estas rutinas fueron:

- Subrutina de entrada y datos.
- Impresión de mensaje y recogida de opción en el menú.
- Subrutina de PULSA UNA TECLA.
- Ordenación alfanumérica.

También vimos un programa de ejemplo para ver cómo funcionaba el programa de ordenación alfanumérica.

El primer programa que vamos a ver en este nuevo tomo va a ser la subrutina encargada de crear los menús en la pantalla. Como el programa funciona por menús, y como hay varios menús en el programa, en vez de tener que dibujar cada uno en su momento realizaremos una rutina que nos permita hacerlos.

Para que esta rutina funcione, hay que pasarle los siguientes datos:

— Cabecera de menú. El mensaje que aparece en la primera línea de la pantalla y que le dice al usuario en qué menú nos encontramos.

— Las distintas opciones del menú.

— Desde qué número hasta qué otro número puede pulsar el usuario para elegir la opción que le interesa. Estos dos valores han de ir almacenados en variables alfanuméricas.

A continuación aparece el programa de esta rutina.



Fig. 1. En nuestro programa habrá muchos menús distintos, por ello necesitamos una rutina para crearlos.



Fig. 2. En este menú, el usuario sólo podrá pulsar un número del 1 al 3.

```

8400 REM
8401 REM *****
8402 REM * SUBROUTINA DE CREACION DE MENUS *
8403 REM *****
8404 REM
8405 CLS
8406 LET X=INT((40-LEN(A$))/2)
8407 LOCATE 1,X
8408 PRINT A$
8409 LOCATE 2,X
8410 FOR Z=1 TO LEN(A$)
8411   PRINT "-";
8412 NEXT Z
8413 PRINT:PRINT
8414 LET X=0
8415 FOR Z=1 TO N
8416   IF X<LEN(A$(Z)) THEN LET X=LEN(A$(Z))
8417 NEXT Z
8418 LET X=INT((40-X)/2)
8419 FOR Z=1 TO N
8420   LOCATE 2*X+4,X
8421   PRINT Z; "-"; A$(Z)
8422 NEXT Z
8423 LET X=1:LET Y=20
8424 GOSUB 8100
8425 CLS
8426 RETURN

```

Programa 1.

Antes de ver y explicar cómo funciona el programa, vamos a ver las modificaciones que hay que hacerle para aquellos ordenadores que no sean ni IBM ni AMSTRAD.

COMMODORE:

```

8405 PRINT "<SHIFT-HOME>"
8407 POKE 214,1:POKE 211,X
8409 POKE 214,2:POKE 211,X
8420 POKE 214,2*Z+4:POKE 211,X
8425 PRINT "<SHIFT-HOME>"

```

MSX:

```

8407 LOCATE X,1
8409 LOCATE X,2
8420 LOCATE X,2*Z+2

```

SPECTRUM:

```

8406 LET X=INT((32-LEN(A$))/2)
8407 PRINT AT 1,X;
8409 PRINT AT 2,X;
8420 PRINT AT 2*Z+1,X;

```

Para que esta rutina funcione tenemos que unirla a la que se dio en el tomo anterior y que hemos llamado: MENSAJE Y RECOGIDA DE OPCION. Esta otra rutina es la encargada de recoger del teclado la opción del usuario.

Las variables que hay que pasarle a la rutina son:

— **A\$**. En ella debemos poner el mensaje que aparecerá en la primera línea y que le dirá al usuario en qué menú se encuentra.

— **N**. En esta variable está almacenado el número de opciones que tiene este menú. Esta variable se utiliza a la hora de imprimir en pantalla dichas opciones.

— **A\$(1) .. A\$(N)**. En este vector van a ir almacenados los distintos textos correspondientes a cada opción. La dimensión de este vector se dará al principio del programa general. Como tenemos **N** posibles opciones, los **N** primeros elementos de dicho vector serán a los que nosotros tenemos que darle un valor antes de llamar a esta rutina.

— **M\$**. En esta variable se almacena la letra o número más alto que el usuario puede introducir por el teclado a la hora de elegir una de las opciones.

— **W\$**. En esta otra variable se almacenará la letra o número más bajo que el usuario puede introducir cuando elija algunas de las opciones.

Antes de empezar con la explicación del programa línea a línea, te recomiendo que introduzcas el programa 2 para ver cómo funciona esta rutina. No se te olvide que tienes que unir este programa con el 1 de este tomo y con el 2 del tomo anterior para que pueda funcionar.

```

10 REM *****
20 REM * PROGRAMA DEMOSTRACION PARA VER *
30 REM * EL FUNCIONAMIENTO DEL PROGRAMA *
40 REM * DE CREACION DE MENUS *
50 REM *****
60 REM
70 CLS
80 LET A$="M E N U   G E N E R A L"
90 DIM A$(6)
100 FOR I=1 TO 6
110   READ A$(I)
120 NEXT I
130 LET W$="1"
140 LET M$="6"
150 LET N=6
160 GOSUB 8400
170 PRINT "TU OPCION HA SIDO LA No. ";A$
180 PRINT
190 PRINT "ESTA OPCION ES:"
200 PRINT A$(VAL(A$))
210 END

```

Programa 2.

TRUCOS Y RUTINAS BASICAS

Las únicas modificaciones que hay que hacer en este segundo programa son:

COMMODORE:

```
70 PRINT "<SHIFT-HOME>"
```

SPECTRUM:

```
90 DIM A$(6,30)
```

Una vez que has ejecutado el programa y que has visto cómo funciona, vamos a ver qué es lo que hace cada línea del programa 1.

Línea 8405. Borra la pantalla.

Línea 8406. Calcula la posición donde habrá que imprimir el mensaje que se encuentra almacenado en A\$ para que éste aparezca centrado en la primera línea de la pantalla. Esto se realiza de la siguiente manera:

- Se resta la longitud de la cadena a imprimir de 40. Esto sirve para cuantos espacios nos sobran en la línea donde se imprimirá el mensaje.

- El número de espacios hallado se divide entre dos. Esto sirve para saber cuántos espacios tienen que estar a la derecha del mensaje. El número de espacios en blanco a la derecha y a la izquierda del mensaje han de ser los mismos.

- Se saca la parte entera del número hallado hasta el momento. Esto puede ocasionar, si el número de caracteres del mensaje es impar, que el texto no quede exactamente centrado.

Con todo esto hemos hallado la columna donde tendremos que imprimir el mensaje que encabezará el menú.

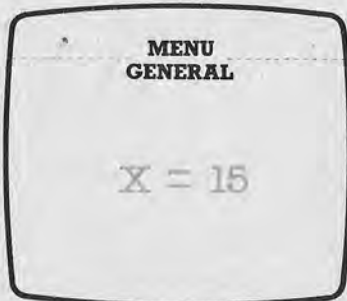


Fig. 3. Con la instrucción $X=INT((40-LEN(A$))/2)$ centramos el mensaje de la cabecera.

Línea 8407. Se coloca el cursor en la primera línea y en la columna que acabamos de hallar.

Línea 8408. Imprimimos el mensaje cabecera del menú.

Línea 8409. Colocamos el cursor en la

segunda línea y en la misma columna donde se empezó a escribir el mensaje.

Línea 8410. Comienza un bucle dentro del cual se imprimirán tantos signos menos (-) como longitud tiene el mensaje cabecera del menú. Esto lo hacemos para que dicho mensaje aparezca subrayado.

Línea 8411. Imprimimos un signo menos (-) en cada vuelta de bucle. El punto y coma después de las comillas sirve para que el cursor no cambie de línea y para que lo que escribamos después aparezca a continuación de lo que acabamos de escribir.

Línea 8412. Aquí se termina el bucle que subraya la cabecera.

Línea 8413. Dejamos una línea en blanco para que las opciones del menú no se junten con la cabecera. El primer PRINT, como no tiene comillas, sirve para decirle al ordenador que, después de subrayada la cabecera, ya puede cambiar de línea. El segundo PRINT es el que deja la línea en blanco.

La parte de programa que viene a continuación se encargará de localizar la opción de menú más larga (con más caracteres) para centrar las opciones con arreglo a su longitud.

Línea 8414. Asignamos a la variable numérica el valor 0. Esta será la variable que va a ir almacenando la cadena que es más larga.

Línea 8415. Empieza un bucle desde uno hasta N, donde se buscará y encontrará la opción de menú con mayor número de caracteres.

Línea 8416. En esta línea se pregunta si X es menor que la longitud de la opción con subíndice Z. En la primera vuelta del bucle, la respuesta a esta pregunta es siempre afirmativa, con lo que la variable X pasa a tomar el valor del número de caracteres de la primera opción del menú. A partir de la segunda vuelta sólo tomará el valor del número de caracteres de una opción cuando ésta sea más larga que la primera o que la última más larga. Con esta línea se consigue que, al final del bucle, se encuentre almacenado en la variable X el valor del número de caracteres de la opción de menú más larga.

Línea 8417. Aquí se termina el bucle.

Línea 8418. Hacemos la misma operación que en la línea 8406, pero ahora es para saber en qué columna imprimiremos todas las opciones del menú para que éstas aparezcan centradas en la pantalla.

Línea 8419. Comenzamos un nuevo bu-

cle dentro del cual imprimiremos todas las opciones del menú.

Línea 8420. Con esta línea vamos a colocar el cursor donde nos interese. En la primera vuelta del bucle lo colocaremos en la línea 6. En la segunda en la 8. En la tercera en la 10 y así sucesivamente. Con este LOCATE (o PRINT AT en el SPECTRUM) conseguimos que la primera opción se separe lo suficiente de la cabecera del menú, y que las distintas opciones de éste estén separadas por una línea en blanco. Así, el menú se hace menos compacto y más fácil de visualizar.

Línea 8421. Imprimimos el número de opción (que coincide con el número de vuelta del bucle), seguido de un guión y de la opción en cuestión.

Línea 8422. Aquí se termina el bucle que se encarga de imprimir todas las opciones del menú.

Línea 8423. Asignamos a la variable X el valor 1 y a la variable Y el valor 20. Si os acordáis del tomo anterior, esto nos va a servir para imprimir el mensaje **INTRODUZCA OPCION** en la columna 1 de la línea 20 cuando llamemos a la rutina de recogida de opción.

Línea 8424. Con el GOSUB 8100 llamamos a la rutina de recogida de opción. No se te olvide que para que el programa funcione, tienes que unirlo con esta rutina que se dio en el tomo anterior.

Línea 8425. Borrarnos la pantalla.

Línea 8426. Y retornamos al menú principal.

Cuando esta rutina se ha ejecutado en su totalidad, podemos saber la opción que ha elegido el usuario con sólo mirar el valor que tiene la variable alfanumérica A\$.



Fig. 4. Esta es la forma de saber lo que ha elegido el usuario.

Una vez que hemos visto y entendido cómo funciona el programa 1, vamos a ver qué es lo que hace el programa 2.

Línea 70. Borrarnos la pantalla del ordenador.

Línea 80. Asignamos a la variable A\$ el mensaje:

MENU GENERAL

éste será el mensaje que aparecerá como cabecera del menú.

Línea 90. Dimensionamos el vector A\$ como un vector de seis elementos. Este es el número de opciones que tiene el menú con el que vamos a trabajar.

Línea 100. Comenzamos un bucle dentro del cual vamos a leer los mensajes que se encuentran almacenados en las líneas DATA. Estos mensajes, que son las distintas opciones posibles del menú general, quedarán almacenadas en el vector que acabamos de dimensionar.

Línea 110. Leemos de la línea DATA la opción que corresponda.

Línea 120. Aquí termina el bucle de lectura de opciones.

Línea 130. Asignamos a la variable alfanumérica el valor «1». Este valor no es el número 1, sino el signo que tiene la forma 1. Esta variable le dirá a la rutina que empieza en la línea 8100 cuál es la tecla con código ASCII más pequeño que el usuario puede introducir por el teclado.

Línea 140. Se asigna a la variable M\$ el carácter con el código ASCII más alto que el usuario puede introducir.

Línea 150. A la variable numérica N se le da el valor 6, que es el número de opciones que va a tener el menú.

Línea 160. Llamamos a la rutina de creación de menús.

A continuación vienen una serie de líneas cuya función es hacer ver al usuario que el programa se ha enterado perfectamente de la opción que ha elegido.

Línea 170. Se imprime el número de opción elegida por el usuario.

Línea 180. Dejamos una línea en blanco.

Línea 190. Se imprime un mensaje.

Línea 200. Y se imprime la opción que se ha elegido. Se imprime el mensaje de dicha opción.

Línea 210. Termina el programa.

A partir de esta línea vienen las líneas DATA que se leen en el programa y que apa-

TRUCOS Y RUTINAS BASICAS

recen en las pantallas como opciones. Te aconsejo que varíes la línea 80 y las líneas DATA para que puedas apreciar bien el funcionamiento de esta rutina. Un ejemplo de lo que podrías hacer es:

```
90 LET A$="MENU DE GRABACION"
220 DATA "GRAB. EN DISCO"
230 DATA "GRAB. EN CINTA"
240 DATA "GRAB. DE ALGUNAS FICHAS EN D."
250 DATA "GRAB. DE ALGUNAS FICHAS EN C."
260 DATA "VERIF. EN DISCO"
270 DATA "VERIF. EN CINTA"
```

El siguiente programa que vamos a ver es una rutina de búsqueda. Esta nos permitirá



Fig. 5. Así aparece la pantalla del ordenador si varías las líneas 90, 220, 230, 240, 250, 260 y 270 como te indica.

buscar una ficha (o un grupo de ellas), bien por sus números de orden, bien por el valor de un campo o ver las fichas que consideramos borradas.

```
8500 REM
8501 REM *****
8502 REM * SUBROUTINA DE BUSQUEDA *
8503 REM *****
8504 REM
8505 CLS
8506 PRINT "B U S C A N D O"
8507 PRINT "===== "
8508 PRINT
8509 IF SW=0 THEN PRINT "Buscando las fichas desde la ";N1;" hasta la ";N2
8510 IF SW=1 THEN PRINT "Buscando las fichas que tengan ";A$;" en el campo ";NC
8511 IF SW=2 THEN PRINT "Buscando fichas borradas"
8512 PRINT
8513 PRINT "ESPERA UN MOMENTO"
8514 FOR Z=1 TO 1000
8515 NEXT Z
8516 CLS
8517 ON SW+1 GOSUB 8526,8537,8567
8518 CLS
8519 PRINT "OPERACION TERMINADA"
8520 PRINT "===== "
8521 PRINT
8522 PRINT "No se han encontrado mas fichas."
8523 LET X=1:LET Y=20
8524 GOSUB 8200
8525 RETURN
8526 REM
8527 REM *** BUSQUEDA POR No. DE FICHA ***
8528 REM
8529 FOR Z=N1 TO N2
8530 LET NF=Z
8531 GOSUB 9000
8532 LET X=1:LET Y=20
8533 GOSUB 8200
8534 CLS
8535 NEXT Z
8536 RETURN
8537 REM
8538 REM *** BUSQUEDA POR CAMPO Y VALOR ***
8539 REM
```

```

8540 LET CC=0
8541 FOR Z=1 TO TT
8542     IF MID$(F$(1),Z,1)=CHR$(254) THEN LET CC=CC+1
8543     IF CC=NC-1 THEN LET Z1=Z:LET Z=TT
8544 NEXT Z
8545 IF CC=NC-1 THEN GOTO 8554
8546 CLS
8547 PRINT "E R R O R"
8548 PRINT "-----"
8549 PRINT
8550 PRINT "Lo siento. El campo ";NC;" no existe."
8551 GOSUB 8200
8552 CLS
8553 RETURN
8554 LET C#=CHR$(254)
8555 FOR Z=1 TO TT
8556     FOR X=1 TO LEN(A$)
8557         IF MID$(F$(Z),X,1)<>MID$(A$,X,1) THEN LET X=LEN(A$)+100
8558     NEXT X
8559     IF X=LEN(A$)+101 THEN GOTO 8565
8560     LET NF=Z
8561     GOSUB 9000
8562     LET X=12:LET Y=20
8563     GOSUB 8200
8564     CLS
8565 NEXT Z
8566 RETURN
8567 REM
8568 REM *** BUSQUEDA DE FICHAS BORRADAS ***
8569 REM
8570 FOR Z=1 TO TT
8571     IF LEFT$(F$(Z),1)<>CHR$(253) THEN GOTO 8579
8572     LET F$(Z)=RIGHT$(F$(Z),LEN(F$(Z))-1)
8573     LET NF=Z
8574     GOSUB 9000
8575     LET X=1:LET Y=20
8576     GOSUB 8200
8577     CLS
8578     LET F$(Z)=CHR$(253)+F$(Z)
8579 NEXT Z
8580 RETURN

```

Programa 3.

Este programa funciona perfectamente en el IBM, AMSTRAD y MSX. Las modificaciones que hay que realizar en el COMMODORE y SPECTRUM son las siguientes:

COMMODORE:

```

8505 PRINT "<SHIFT-HOME>"
8516 PRINT "<SHIFT-HOME>"
8518 PRINT "<SHIFT-HOME>"
8534 PRINT "<SHIFT-HOME>"
8546 PRINT "<SHIFT-HOME>"
8552 PRINT "<SHIFT-HOME>"
8561 PRINT "<SHIFT-HOME>"
8577 PRINT "<SHIFT-HOME>"

```

SPECTRUM:

```

8517 GOTO 8526(SW=0)+8537(SW=1)
+8567(SW=2)
8542 IF F$(1,Z)=CHR$(254) THEN LET
      CC=CC+1
8557 IF F$(Z,X)<>A$(X) THE LET
      X=LEN(A$)+100
8571 IF F$(Z,1)<>CHR$(253) THEN GOTO
8579
8572 LET F$(Z)=F$(Z,2 TO)

```

La elección del tipo de búsqueda vendrá dada por el menú general. Se le presenta-

TRUCOS Y RUTINAS BASICAS

rá al usuario un menú con tres opciones. Estas serán:

1. Búsqueda por número de ficha.
2. Búsqueda por campo.
3. Búsqueda de fichas borradas.

Según la opción que se elija, una variable numérica **SW** tomará el valor 0, 1 ó 2. Según este valor, realizaremos una de las tres funciones.

Este programa no puede ser ejecutado todavía por varias razones:

1. Todavía no tenemos la rutina de impresión de fichas.
2. No hemos metido ninguna ficha en el fichero.
3. No hemos definido el fichero.

Aun así, vamos a ver su funcionamiento línea a línea:

Línea 8505. Borrarnos la pantalla.

Línea 8506. Se imprime el mensaje: BUSCANDO.

Línea 8507. Se subraya el mensaje anterior.

Línea 8508. Y dejamos una línea en blanco.

Línea 8509. Si se eligió la búsqueda por el número de orden, se da un mensaje avisándolo y diciendo desde qué ficha hasta qué otra ficha se va a realizar la búsqueda.

Línea 8510. Si se eligió la búsqueda por número de campo, se avisa, se dice qué número de campo y qué hay que buscar.

Línea 8511. Por último, si se querían ver las fichas que se encuentran borradas, se imprime una mensaje informando de ello.

Línea 8512. Dejamos una línea en blanco: .

Línea 8513. Y se le dice al usuario que espere un momento.

Líneas 8514 y 8515. Se realiza un bucle de retardo para que lo último que hemos impreso no se borre de la pantalla

Línea 8516. Borrarnos la pantalla para comenzar la búsqueda.

Línea 8517. Según el valor que tenga la variable numérica **SW**, nos vamos a un número de línea o a otro. Esto es así porque depende del tipo de búsqueda que queramos, el ir a una parte o a otra de la rutina.

Línea 8518. Una vez que se han visualizado todas las fichas que queríamos, borramos la pantalla.

Líneas 8519 hasta 8522. Avisamos al

usuario de que la operación está terminada y de que no hemos encontrado más fichas.

Líneas 8523 y 8524. Llamamos a la rutina de pulsa una tecla, y decimos que el mensaje se coloque en la columna 1 de la fila 20.

Línea 8525. Volvemos al programa principal.

A partir de este lugar se encuentran las tres rutinas que necesitamos para realizar la búsqueda de la manera que queramos. La primera rutina realiza la búsqueda por el número de orden de las fichas.

Línea 8529. Comienza un bucle desde la primera a visualizar hasta la última, dentro del cual se verán todas las fichas que hemos podido ver.

Línea 8530. Se asigna a la variable **NF** el valor de la variable índice del bucle **Z**. **NZ** es la variable que necesita la rutina de visualización de fichas para imprimir la que nosotros deseemos.

Línea 8531. Llamamos a la subrutina que nos imprime las fichas en pantalla o en impresora.

Líneas 8532 y 8533. Imprimimos el mensaje de pulsa una tecla en la primera columna de la fila 20.

Línea 8534. Borrarnos la pantalla.

Línea 8535. Y continuamos con el bucle hasta haber visto todas las fichas que queríamos.

Línea 8536. Volvemos a la rutina principal para decir que no hay más fichas que mostrar y volver al programa principal.

Con el consiguiente grupo de líneas conseguimos visualizar todas las fichas que tengan un cierto valor en un cierto campo. El grupo de líneas que van desde la número 8540 hasta la 8553 son iguales que las que vimos en el programa 4 del tomo anterior. Su función es comprobar que el campo **NC** existe en este fichero. El resto de las líneas nos permiten la visualización de todas las fichas que contengan dicho campo.

Línea 8554. Asignamos a la variable alfanumérica **C\$** el carácter 254. Este carácter, como todos recordaréis, era el separador y delimitador de campos.

Línea 8555. Comenzamos un bucle que va desde la primera ficha hasta la última. Dentro de él sólo veremos las que cumplan la condición por nosotros impuesta.

Línea 8556. Comienza otro bucle que va desde 1 hasta la longitud del mensaje que va

mos a buscar. Este bucle sirve para comparar letra a letra el valor del campo con lo que buscamos. Tal y como hemos hecho la rutina, y gracias a este bucle, podemos decirle al programa que nos busque, por ejemplo, todas aquellas fichas que al principio del campo número 3 tengan la secuencia de letras **Frc** o cualquier otra.

Línea 8557. Se comprueba si cada carácter de A\$ coincide con cada carácter del campo NC de F\$(Z). En caso negativo, nos salimos del segundo bucle. En caso positivo, continuamos con él.

Línea 8558. Aquí termina el segundo bucle.

Línea 8559. Si el campo era distinto de lo que estábamos buscando, nos vamos a mirar la siguiente ficha.

Línea 8560. Si el campo era igual, en-

tonces asignamos a NF el número de ficha que lo contiene.

Línea 8561. Y la visualizamos.

Líneas 8562 y 8563. Imprimimos el mensaje: PULSA UNA TECLA.

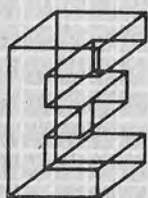
Línea 8564. Borrarnos la pantalla.

Línea 8565. Y continuamos con el bucle en busca de más fichas.

Línea 8566. Retornamos a la rutina principal para decirle al usuario que no se han encontrado más fichas.

El siguiente grupo de líneas es el encargado de mostrarnos todas las fichas que hemos borrado del fichero, pero no de la memoria. Como no hemos explicado todavía cómo vamos a enmascarar este tipo de fichas, dejaremos la explicación de esta última parte del programa para más adelante.

■ Palancas de mando o joysticks



S cada vez más frecuente en los ordenadores personales disponer de instrumentos de comunicación de órdenes que no utilizan el teclado. Los más comunes son las palancas de mando o joysticks, que permiten enviar al

programa la indicación de una acción inmediata o de una variable proporcional a la posición ocupada en un momento determinado. También suelen incluir algún botón que permita la activación de una parte del programa o más comúnmente el disparo de algún tipo de arma sofisticada. Vamos a describir algunos tipos de palancas de mando y de las interfaces necesarias en los ordenadores para su empleo en juegos y programas de diseño gráfico.

En primer lugar hay que distinguir los tipos de palancas, desde el punto de vista electrónico:

- Binarias, que indican la posición mediante contactos. Pueden ser contactos específicos para la palanca o bien simular los contactos propios del teclado.

- Proporcionales, que indican la posición mediante el valor de una resistencia o tensión, proporcionales de acuerdo con una escala apropiada.

Según el tipo de señal eléctrica obtenida deberemos utilizar un tipo diferente de interfaz y, por supuesto, de programa, para conseguir la captación de lo que se desea hacer en el juego o aplicación.

Desde el punto de vista de la apariencia exterior y por consiguiente del tipo de aplicaciones para los que puede usarse podemos distinguir:

- Palanca con forma adaptada a la mano, con botones para disparo rápido. Existen modelos para sujeción a la mesa o para soporte en una mano. Deben ser robustos, pues han de soportar miles de acciones contra feroces enemigos. Poseen siempre auto-centrado. Como opción suelen disponer de disparo automático, para reducir la fatiga del usuario.

- Bola con contactos internos y botones próximos. Es una robusta variante que permite el control con muy poco desplazamiento de la mano.

- Palanca con poca resistencia al desplazamiento, con botones en caja base para indicación de acciones. Pueden disponer de autocentrado y de ajuste del valor central de las resistencias.

En la figura se ven algunos de los tipos característicos de cada uno.

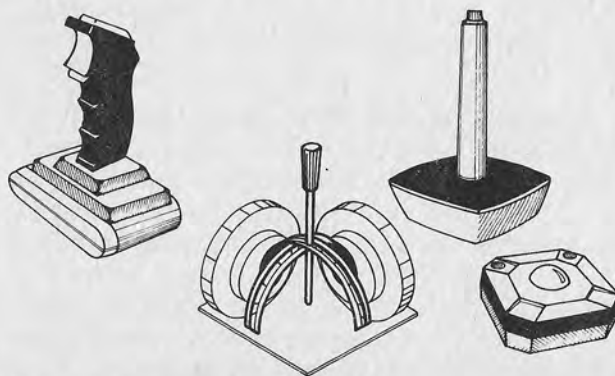


Fig. 1. Tipos de palancas de mando.

La parte eléctrica de las palancas de mando es un proyecto fácil de construir por nuestros medios y nos permitirá dotar a nuestro ordenador de comunicación distinta a la del teclado. Para hacerlo robusto mecánicamente deberemos, además, dotarle de alguna caja y mando apropiado para la aplicación a la que lo queramos destinar.

Empezaremos por describir eléctricamente cada uno de los tipos y las interfaces necesarias para algunos ordenadores.

■ Palancas mando de tipo binario

El circuito necesario para realizar la comunicación de variables binarias ya lo hemos presentado en los primeros capítulos. Consiste simplemente en un puerto de entrada direccionable en una zona libre de direcciones. Cada bit de entrada aparecerá como uno de los bits del octeto correspondiente a la lectura de la dirección. Según el tipo de ordenador deberá ser validada la dirección con las señales indicativas de instrucción de entrada-salida o de acceso a memoria y alguna señal de temporización.

Para la indicación de movimientos en las dos direcciones y los dos sentidos necesitaremos 4 bits. Además, para los botones de disparo emplearemos una o dos entradas más. La dirección sobre la que aparecerán los contactos de dirección de la acción y los de disparo puede ser la misma, pues normalmente el programa funciona por exploración del puerto de entrada, para conocer si ha habido algún cambio desde la lectura anterior.

Normalmente es suficiente con disponer de una sola palanca de mandos, pues jugaremos contra el ordenador, pero puede ser conveniente disponer de dos palancas, para juegos de competición con otro usuario o para comunicar dos juegos de variables, como es el caso de los simuladores de vuelo.

Los juegos existentes en el mercado requieren al principio de la primera jugada la indicación del tipo de interfaz usada. Al no existir normalización sobre este tipo de aparatos cada constructor ha diseñado la interfaz más conveniente para sus propósitos. Por eso el diseño apropiado para cada uno dependerá del tipo de programas que desee utilizar. La verdad es que las diferencias entre unos y otros fabricantes son pequeñas, pero no es fácil el realizar un diseño que además de económico

pueda adaptarse a cualquiera de los tipos conocidos. Por ello vamos a describir los que son más usuales y que pueden construirse con facilidad.

■ Ejemplo de palanca de contactos

Podemos construir una palanca de mando, mediante cuatro teclas y una barra vertical que pueda hacer presión sobre ellas, inclinandola en la dirección adecuada. El esquema puede ser construido con elementos muy económicos y servir para probar la interfaz.

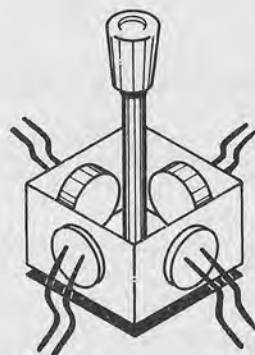


Fig. 2. Ejemplo de palanca de mandos con pulsadores.

■ Ejemplos para ZX-Spectrum

Empezaremos por los tipos más conocidos para juegos del Spectrum.

Tipo Kempston. Utiliza una de las direcciones de puerto libres que se direcciona mediante el bit A5. Por tanto, obtenemos la decodificación de la manera habitual mediante -IORQ, -A5 y -RD. El signo negativo es para indicar que se utiliza la señal complementada. Podríamos utilizar la barra encima del nombre, pero es un poco más complicado hacerlo con un procesador de texto sencillo en un ordenador personal.

La señal de decodificación activa un amplificador de bus de tres estados que estará conectado de acuerdo con la figura 3.

La tarjeta resulta sumamente sencilla, solamente necesita 2 chips. Si ya hemos realizado la tarjeta de ampliación de puertos, descrita en los fascículos primero y segundo, se puede conectar directamente la palanca de mandos a uno de los puertos de entrada incluidos en ella. El conector deberá ser del tipo apropiado para la tarjeta. En las palancas ori-

EL TALLER DEL HARDWARE

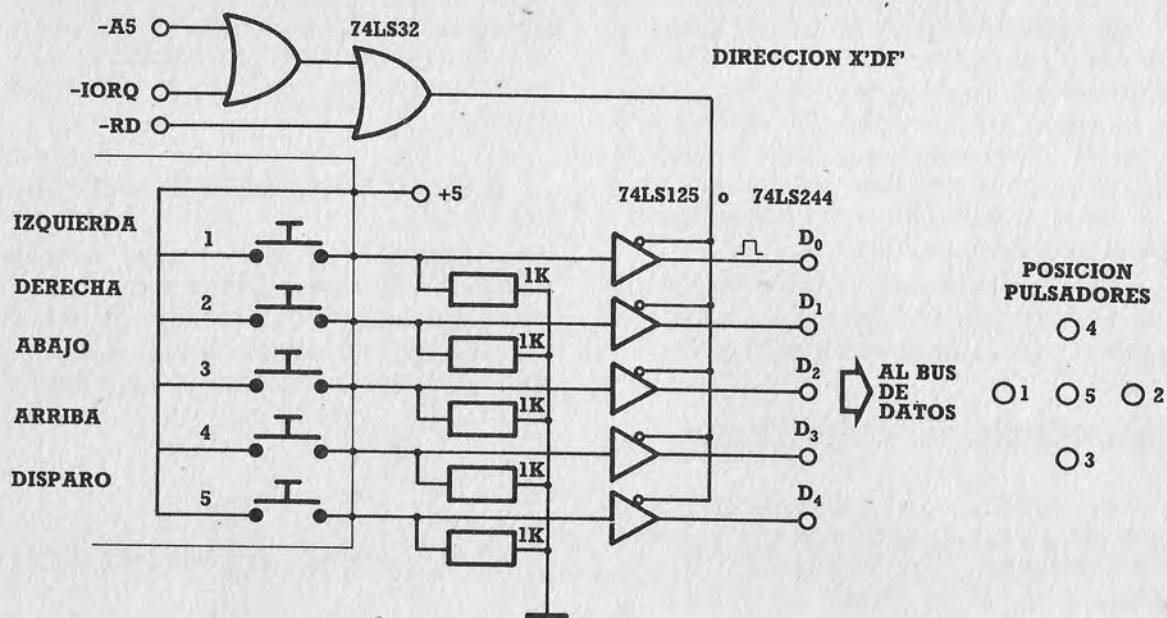


Fig. 3a. Interfaz tipo Kempston para Spectrum.

ginales se emplea un conector de 9 pines con la asignación que se indica, pero para cada uno debe adoptar la solución más conveniente a sus posibilidades. Lo importante es utilizar la dirección indicada, para que sea compatible con los juegos ya existentes. Además, debe también coincidir el nivel que se interpreta como activo. En reposo los niveles normales están en 0, al activarse al interruptor que se cierra hace subir la señal a 1, permaneciendo los demás en 0. La correspondencia entre contactos y la dirección y sentido se indica en la figura.

Tipo Sinclair. Utiliza direcciones correspondientes al teclado, para lo cual decodifica posiciones asignadas a las filas superiores. Las señales decodificadas son: -IORQ, -RD, -A11 y -A0 para la entrada 1 y la dirección -A12 en lugar de la -A11 para la entrada 2. El nivel de reposo de las entradas es 1, al activarse cualquier interruptor hace bajar su entrada a 0. El juego de palancas exteriores y los contactos es compatible con el tipo Kempston, pero cambia la asignación de la conexión común en la interfaz, pues en el primero se conecta a +5 voltios y en el segundo a masa.

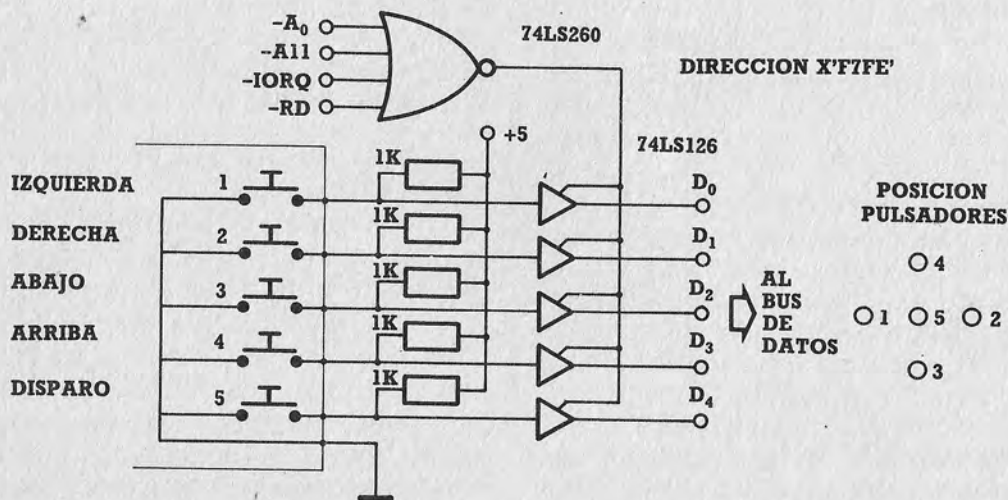


Fig. 3b. Interfaz tipo Sinclair para Spectrum.

La utilización es exactamente la misma que la anterior.

El Commodore-64 posee internamente interfaz para 4 pulsadores y dos entradas de resistencia variable, además de entrada de disparo.

■ Circuito de disparo automático

Para producir el efecto de pulsación muy rápida del botón de disparo, podemos montar un circuito que envíe una señal de la misma forma que el contacto manual, pero mucho más de prisa. Claro está que el programa podría "olérselo" y en algún caso "hacerse el sordo" si ve que responde muy de prisa para un ser humano. De todas maneras, como el circuito es muy simple podremos montarlo en el caso de que hagamos un montaje para esta aplicación.

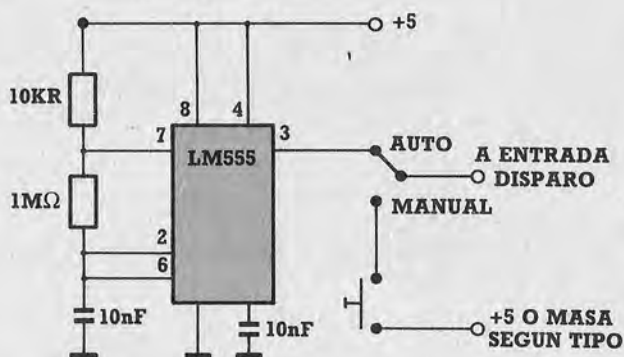


Fig. 4. Circuito de disparo automático.

El circuito consiste en un oscilador basado en un LM555 y que está oscilando siempre a una frecuencia fija. Al poner el conmutador de disparo en posición Automático se están enviando ráfagas continuamente a la entrada de disparo. Si es necesario que la entrada de disparo esté desactivada al comenzar la acción podemos hacer que el pulsador de disparo sea en realidad un conmutador que pasa de la conexión pulso a pulso a la de ráfagas, pero empezando siempre en reposo.

■ Palancas de mando de tipo proporcional

Para juegos sofisticados y diseño gráfico es necesario disponer de la posibilidad de

indicación de valor absoluto de un ángulo o coordenada. La variable eléctrica más asequible para ser utilizada de forma mecánica es una resistencia asociada a la posición de la mano. Según que el valor de la resistencia proporcione una tensión o una frecuencia dependiente de ella, tendremos que utilizar interfaz de conversión analógico/digital o simplemente una medida de período o número de pulsos.

■ Palanca de mando de resistencias

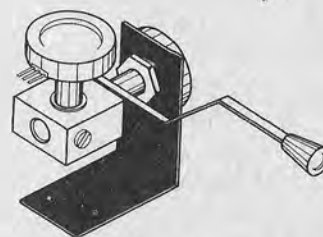


Fig. 5. Palanca de mando de resistencias.

En la figura se muestra una disposición posible de dos resistencias variables para ser utilizadas como palanca de mando. La construcción puede parecer complicada, pero es sumamente simple si se tienen en cuenta las siguientes indicaciones:

- Utilizar resistencias variables de tipo lineal y eje de plástico.
- Realizar un bloque de plástico con dos taladros perpendiculares del diámetro de los ejes de las resistencias variables.
- Utilizar como anclaje una de las resistencias variables (A), dejando los terminales hacia abajo, para poder realizar las conexiones.
- Utilizar la tuerca de anclaje para calibrar la posición central.
- Los ejes de las resistencias variables irán sujetos al bloque de plástico mediante tornillo prisionero o mediante un pegamento.
- La palanca de mando se sujetará a la tuerca de la resistencia variable móvil (B).
- Utilizar hilos flexibles torcidos para las conexiones.

Las palancas de mando comerciales suelen poseer un recorrido limitado por la carcasa del equipo. El ángulo normalmente admisible no pasa de 90 grados. Como las resistencias variables poseen un ángulo de trabajo de 0 a 300 grados se presenta un problema que

EL TALLER DEL HARDWARE

es necesario tener en cuenta: el valor leído en los extremos de posición de las resistencias no corresponde con los límites de los valores de las resistencias.

Para corregir este defecto podemos hacer dos cosas: cambiar la correspondencia lectura-valor o variar algún parámetro eléctrico para que facilite la lectura en los extremos. El cambio de correspondencia lectura-valor solamente es posible en programas que podamos cambiar y, por tanto, no es aplicable a los que ya existen. El cambio de algún parámetro del circuito es posible (añadir una resistencia o cambiar un condensador), pero de forma más costosa e incómoda.

Las interfaces de tipo proporcional de Commodore-64 y la tarjeta adaptadora de juegos del IBM-PC son del tipo de resistencia variable, que controla un oscilador monoestable.

Es muy conveniente que las palancas tengan autocentrado opcional. Para incluirlo en el montaje que se indica, se pueden añadir unos muelles que fuerzan suavemente la posición de los ejes. Para que se pueda trabajar sin autocentrado, es suficiente colocar los muelles en posición de máxima apertura, para que no toquen al eje central. Los sistemas de autocentrado no trabajan bien si el ángulo de utilización es mayor que 90 grados.

Veamos cómo podemos utilizar en la práctica las palancas de mandos de tipo proporcional.

Interfaz de tipo tensión

Se alimentan mediante tensión continua estabilizada y la lectura consistirá simplemente en la conversión de la magnitud analógica tensión en el código correspondiente. La interfaz necesaria es cualquiera de las indicadas para la conversión analógico-digital propuestas en fascículos anteriores. Utilizando la tarjeta de ampliación de puertos de entrada-salida, la lectura se introducirá por uno de los

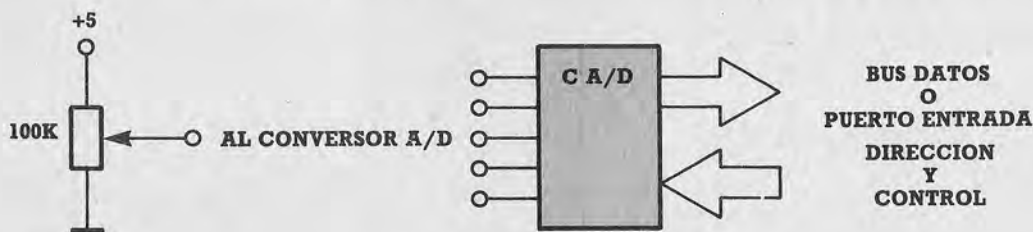


Fig. 6. Interfaz tipo tensión.

puertos de entrada. El control se hará por el puerto de salida. Si se utiliza el conversor A/D propuesto, con 8 entradas analógicas posibles, podremos emplear hasta 4 juegos de palancas de mando dobles, como la indicada en la figura anterior.

Interfaz de tipo frecuencia

La resistencia variable utilizada como indicador de posición determina la frecuencia de un oscilador, realizado con un circuito de tipo LM555 o sus derivados con 2 circuitos (LM556) o con 4 (LM558). El oscilador está siempre en funcionamiento, por lo que el programa solamente debe determinar la frecuencia de oscilación, midiendo el número de oscilaciones en un período fijo de tiempo.

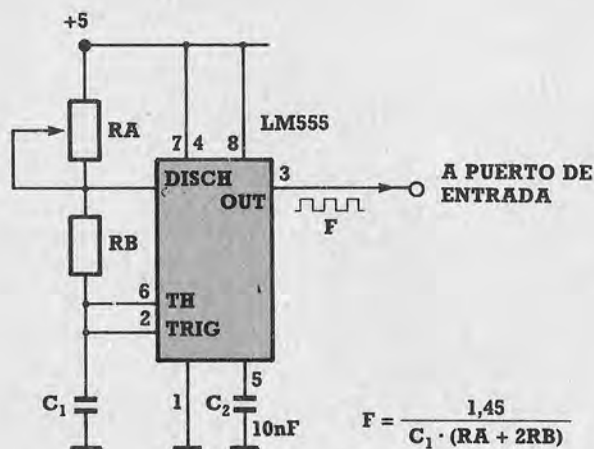


Fig. 7. Interfaz de tipo frecuencia.

Interfaz de tipo período

De forma muy parecida al caso anterior, puede medirse con un circuito muy similar el tiempo de mantenimiento en modo monoestable de un circuito realizado utilizando un LM555. El circuito requiere ser activado cada

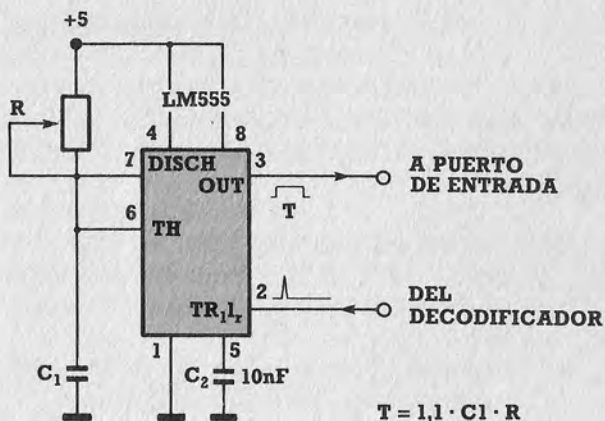


Fig. 8. Interfaz de tipo periodo.

vez que se va a efectuar una medida. Para activar la lectura se asocia el pulso de arranque al decodificador de dirección, válido con la señal de lectura del bus. El programa que realice la lectura deberá contar internamente con un bucle de instrucciones de duración conocida, para poder determinar duraciones de tiempo. El programa medirá el tiempo durante el cual está activada la entrada correspon-

diente a la resistencia variable de la magnitud a medir.

Un ejemplo de interfaz de este tipo es la tarjeta del adaptador de juegos del IBM-PC.

Utilización de las palancas de mando

Además de para acribillar a los enemigos espaciales, las palancas de mando binarias pueden utilizarse para otras acciones creativas. La interfaz de programación es bien sencilla, solamente se requiere leer el puerto correspondiente y actuar en el sentido adecuado a la "tecla" pulsada. Las acciones asociadas pueden ser de lo más variado, pero lo que sí puede ser de interés es cómo utilizar la indicación de dirección y sentido. Por ejemplo, podríamos hacer que el incremento que se utilizara en el movimiento del cursor indicativo de la posición fuera variable, según una tecla de función, mejor todavía, según el tiempo que llevemos pulsando una determinada dirección. Así, podríamos conseguir un comportamiento adaptativo incrementalmente y

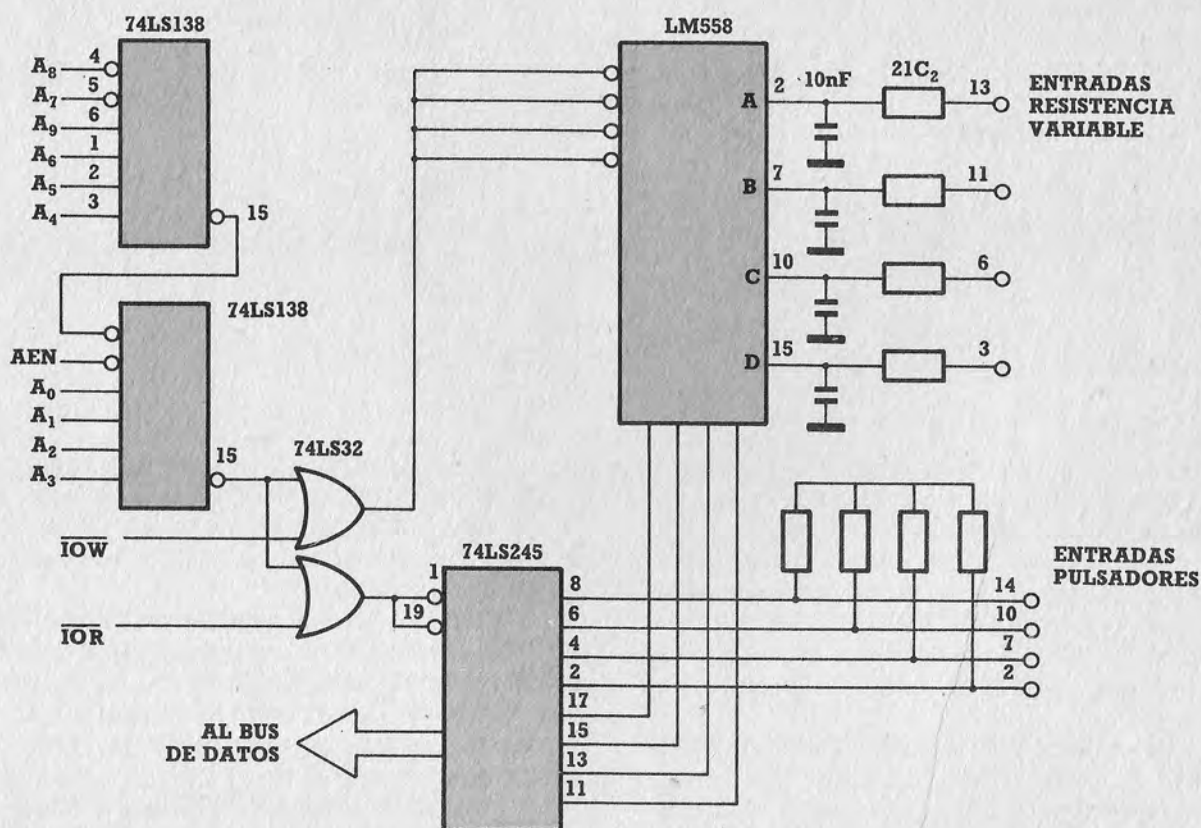


Fig. 9. Tarjeta del adaptador de juegos IBM-PC.

EL TALLER DEL HARDWARE

con una finura de resolución variable según las conveniencias.

■ Circuito integrado LM555

Este económico circuito tiene múltiples aplicaciones en todo taller de electrónica o informática, en su variante hardware. Se le puede emplear como oscilador básico, controlado por cualquier tipo de magnitud a la que se quiera asociar la frecuencia de oscilación.

El esquema de bloques del circuito nos indica las entradas por las que podemos programarlo y las salidas de las que se dispone.

sión de control) permite utilizarlo en conversión tensión-período, pues hace variar los niveles de comparación y, por tanto, los tiempos de carga y descarga del condensador. En realidad, la señal generada es de tipo modulación de ancho de pulso.

La entrada T (Trigger = disparo) permite fijar el nivel o las condiciones de descarga del condensador. Igualmente la entrada Reset sirve para poner a cero el biestable de salida, digitalmente.

Las salidas son de dos tipos: una digital, para utilización como señal compatible TTL y otra de transistor en colector abierto, que permite la conmutación de señales de tensiones superiores.

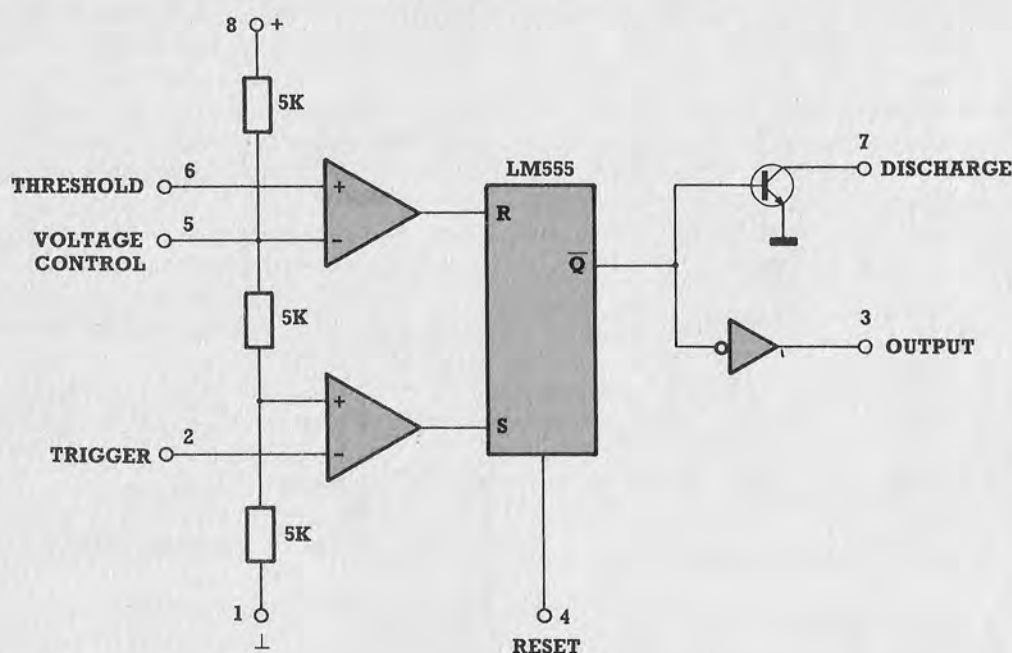


Fig. 10. Esquema de bloques LM555.

Presenta dos amplificadores comparadores a la entrada, que tienen una de sus entradas conectada a un divisor resistivo. El divisor resistivo crea unos niveles de comparación dependientes de la tensión de alimentación, por lo que puede garantizarse gran estabilidad si todas las constantes de tiempo de los circuitos utilizados toman la tensión del mismo sitio.

La entrada TH (Threshold = umbral) fija el nivel de comparación utilizado para realizar la operación de oscilación estable, mediante la carga y descarga de un condensador.

La entrada CV (Voltage control = Ten-

En las figuras adjuntas se muestran circuitos de aplicación del LM555 a la generación de señales utilizables en montajes de lectura de magnitudes asociadas con interfaces de ordenadores personales.

Como oscilador **estable**: se deben utilizar tres componentes externos para fijación de los tiempos de la oscilación. Las resistencias RA y RB determinan las constantes de tiempo de carga del condensador, hasta alcanzar el nivel de 2/3 de la tensión de alimentación. Después se descarga hasta 1/3 de la tensión de alimentación a través de RB. Mediante los valores de RA y RB se puede variar

la frecuencia de oscilación y el tiempo en estado 1 de la salida. La fórmula en la figura indica la frecuencia de oscilación y el tiempo en estado 1. El condensador C1 determina el rango de frecuencias. El condensador C2 solamente se emplea para garantizar la oscilación estable (ver figura 7).

Como oscilador **monoestable**: la duración de la salida activada está determinada por los valores de la resistencia RA y el condensador C1. Los terminales de umbral y descarga se fijan al mismo nivel, por lo que al al-

canzarse el condensador se descargará hasta que vuelva a dispararse de nuevo (ver figura 8).

Como modulador de ancho de pulso, se utiliza la entrada de tensión de control para determinar el nivel de comparación. El nivel de comparación determina a su vez la duración en estado monoestable, para unos valores de los componentes RA y C1. Empleado en combinación con un oscilador estable permite la generación de trenes de pulsos de frecuencia fija y ancho de pulso variable. Se emplea en fuentes de alimentación conmutadas.

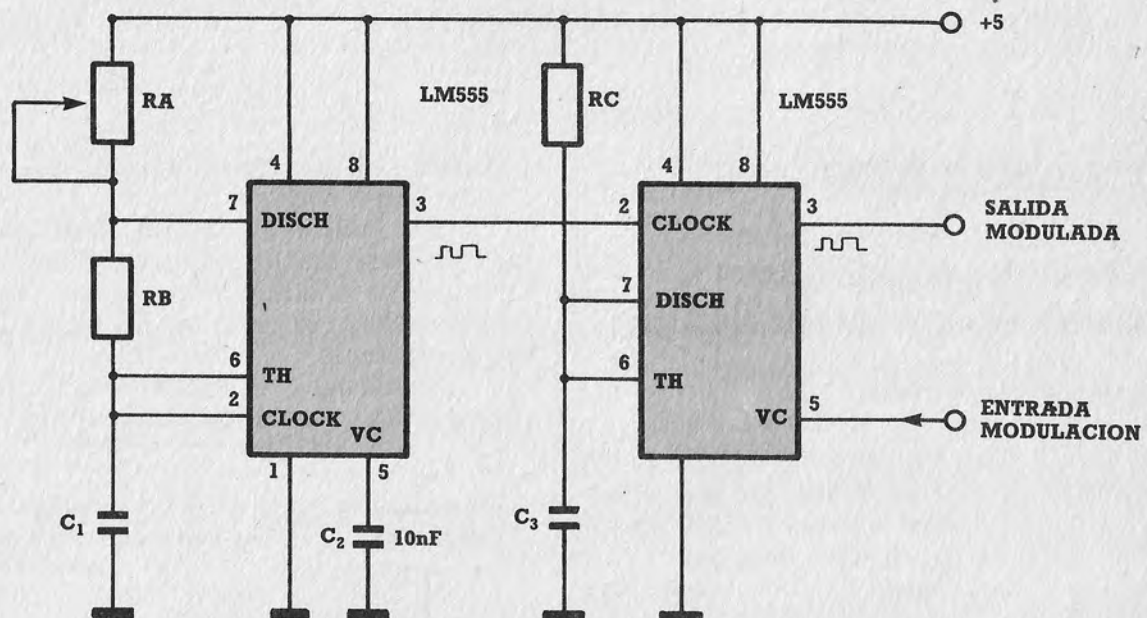


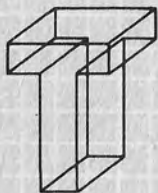
Fig. 11. Circuito de aplicación de LM555.

APRENDER CON EL ORDENADOR

NATURALEZA Y TECNOLOGIA

■ Cálculo de probabilidades. Simulación de experimentos aleatorios

1. Sucesos equiprobables.



RADICIONALMENTE se ha considerado que sólo se podían estudiar científicamente aquellos fenómenos que siempre daban los mismos resultados en igualdad de condiciones. Este es el caso del lanzamiento de un objeto desde un tercer piso: siempre caerá al suelo.

Existe otro tipo de fenómenos en los cuales los resultados no son predecibles. Son los denominados **sucesos aleatorios** o de azar. El lanzamiento de una moneda al aire, o los resultados de la lotería son ejemplos de sucesos aleatorios, puesto que no se pueden predecir con absoluta seguridad.

En esta ocasión, vamos a simular un experimento aleatorio: el lanzamiento de uno o dos dados.

Para ello, vamos a aprovechar una función del lenguaje BASIC que sirve para obtener números "aleatorios". Se trata de la función RND, cuyo resultado es un número comprendido entre 0 y 1, sin llegar nunca a este último valor. Si queremos obtener números entre 1 y 6, tendremos que realizar el siguiente proceso:

1. Multiplicar la función RND por 6

para obtener números comprendidos entre 0 y 6, sin llegar nunca a valer 6.

2. Aplicar la función INT al producto anterior para obtener números enteros entre 0 y 5, ambos inclusive. Ello se debe a que esta función calcula la parte entera de un número sin redondearlo.

3. Sumar 1 al resultado anterior para obtener números enteros comprendidos entre 1 y 6.

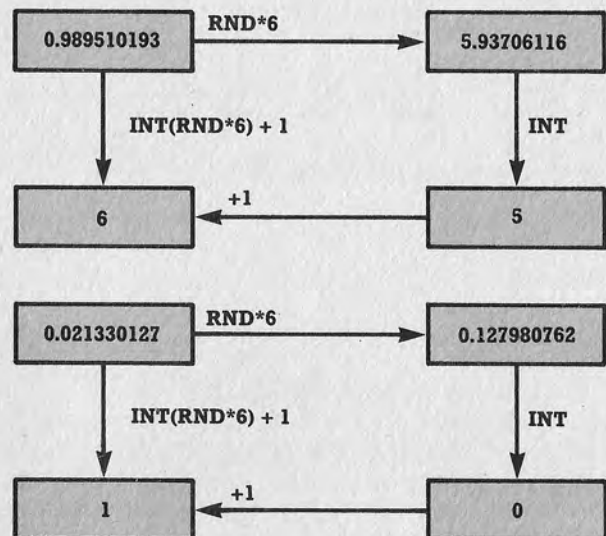


Fig. 1. Ejemplos de la obtención de números aleatorios entre 1 y 6.

En el programa, además, introducimos un contador con subíndice, C(I), que dependerá del número obtenido aleatoriamente. Si se obtuviese un 2, por ejemplo, el contador C(2) se incrementaría en una unidad, mientras

que si fuera un 6 el resultado, el contador que aumentaría sería C(6) y así sucesivamente. De esta manera obtendremos la frecuencia de los resultados obtenidos.

Dado que la función RND no es estrictamente una forma de generar números aleatorios, incluimos la función RANDOMIZE TIMER, propia del IBM, que genera una serie de números dependiendo del reloj del ordenador. Esta función se expresa de modo diferente en otros equipos, tal y como indicamos en las modificaciones.

```

10 REM *****
20 REM * PROGRAMA DE CALCULO *
30 REM * DE FRECUENCIAS *
40 REM * AL LANZAR UN DADO *
50 REM *****
60 REM
70 REM *****
80 REM * IBM,MSX,COMMODORE,SPECTRUM,AMSTRAD *
90 REM *****
100 CLS
110 DIM C(6):LET T=0
120 INPUT "NUMERO DE TIRADAS";N
130 CLS
140 PRINT "E S P E R A   U N   M O M E N T O"
150 FOR A=1 TO N
160 RANDOMIZE TIMER
170 LET I=INT(RND*6)+1
180 LET C(I)=C(I)+1
190 LET T=T+1
200 NEXT A
210 CLS
220 PRINT "NUMERO","FRECUENCIA"
230 PRINT:PRINT
240 FOR I=1 TO 6
250 PRINT I,C(I);"/";N; "=" ;INT(C(I)/
    N*100+.5)/100
260 NEXT I
270 PRINT:PRINT
280 PRINT "TOTAL:","T;"/";N

```

Programa 1.

■ Modificaciones para otros equipos SPECTRUM

160 RANDOMIZE 0

MSX y COMMODORE

No poner la línea 160.

170 LET I=INT(RND(1)*6)+1

Además, en el Commodore, sustituir las instrucciones CLS por PRINT CHR\$(143)

AMSTRAD

160 RANDOMIZE TIME

El número de tiradas del dado es una variable N, que dejamos a la elección del lector. Recomendamos que éste experimente con el número de tiradas. Cuanto mayor sea este número, más se acercarán los resultados al

concepto de **probabilidad**. Esta tendencia constituye la ley del azar, que dice: «Al aumentar el número de pruebas de un experimento aleatorio, la frecuencia de un suceso tiende a estabilizarse alrededor de un número que coincide con su probabilidad».

Si efectuamos el lanzamiento de un dado, existirá la misma probabilidad de obtener un 1, o un 2, etc.

Sin embargo, podríamos lanzar un dado al aire tres veces y obtener tres seises. Si el número de experimentos es mucho mayor, la frecuencia relativa de obtener un seis será de 1/6, según la fórmula:

$$p(A) = \frac{m}{n}$$

donde:

m = número de casos favorables (1)

n = número de casos posibles (6)

Debemos hacer notar al lector que hemos empleado un "truco" para obtener un número redondeado a dos decimales, válido para cualquier tipo de ordenadores:

$$\text{INT} (C(I)/N*100+.5)/100$$

Para redondear un número a dos decimales, lo multiplicamos por 100, le sumamos 0.5, extraemos la parte entera y lo dividimos por 100.

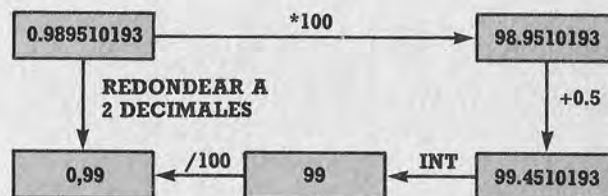


Fig. 2. Redondeo de un número a dos decimales, válido para cualquier ordenador.

Como curiosidad hemos construido un programa en el que se simula el lanzamiento de un dado de manera gráfica. Para ello se construye un dado formado por puntos y un contorno cuadrado.

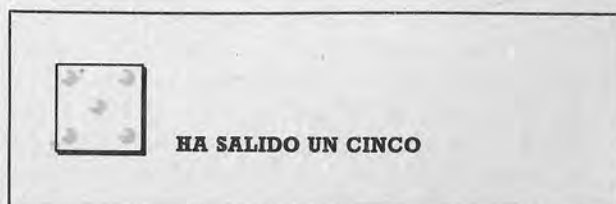


Fig. 3. Simulación gráfica del lanzamiento de un dado.

APRENDER CON EL ORDENADOR

Los puntos se sitúan en las coordenadas adecuadas, según sea el resultado obtenido. Utilizamos las variables L(I,J) y C(I,J) para guardar las coordenadas de las caras de los dados.

```

10 REM *****
20 REM * SIMULACION DEL LANZAMIENTO DE UN DADO *
30 REM *****
40 REM
50 REM *****
60 REM * VALIDO PARA IBM *
70 REM *****
80 CLS
90 SCREEN 1
100 DIM F(6):DIM P(6)
110 DIM L(6,6):DIM C(6,6)
120 REM *****
130 REM * LECTURA DE COORDENADAS DEL DADO *
140 REM *****
150 FOR I=1 TO 6
160 FOR J=1 TO 6
170 READ L(I,J),C(I,J)
180 NEXT J
190 NEXT I
200 REM *****
210 REM * TIRADA DEL DADO *
220 REM *****
230 FOR A=1 TO 10
240 GOSUB 1000
250 RANDOMIZE TIMER
260 LET I=INT(RND*6)+1
270 LET F(I)=F(I)+1
280 FOR B=1 TO 6
290 LOCATE L(I,B)+13,C(I,B)+7:PRINT " "
300 NEXT B
310 PRINT:PRINT:PRINT "HA SALIDO UN:";I
320 FOR Z=1 TO 2000:NEXT Z
330 CLS
340 NEXT A
350 FOR I=1 TO 6
360 PRINT "EL NUMERO ";I;" HA SALIDO ";F(I);" VECES"
370 NEXT I
380 END
1000 REM *****
1010 REM * CONTORNO DEL DADO *
1020 REM *****
1030 LINE (3,103)-(80,103)
1040 LINE (6,103)-(80,130)
1050 LINE (80,130)-(53,130)
1060 LINE (53,130)-(53,103)
1070 RETURN
2000 REM *****
2010 REM * DATOS *
2020 REM *****
2030 DATA 2,2
2040 DATA 1,1,3,3
2050 DATA 1,1,2,2,3,3
2060 DATA 1,1,1,3,3,1,3,3
2070 DATA 1,1,1,3,2,2,3,1,3,3
2080 DATA 1,1,1,2,1,3,3,1,3,2,3,3

```

Programa 2.

Modificaciones para otros equipos AMSTRAD

```

90 NO PONER
250 RANDOMIZE TIME
290 LOCATE L(I,B)+5,C(I,B)+15:PRINT " "
1025 PLOT 75,160
1030 DRAWR 55,0
1040 DRAWR 0,-55

```

```

1050 DRAWR -55,0
1060 DRAWR 0,55

```

SPECTRUM

```

90 NO PONER
250 RANDOMIZE 0
290 PRINT AT L(I,B)+10,C(I,B)+5;" "
1025 PLOT 44,88
1030 DRAW 30,0
1040 DRAW 0,-30
1050 DRAW -30,0
1060 DRAW 0,30

```

2. Sucesos no equiprobables.

Cuando se lanza un dado al aire, la probabilidad de obtener un 1 es la misma que la de obtener un 5. No sucede lo mismo cuando se lanzan dos dados y se anota el resultado de la suma. La probabilidad de obtener un 3 es mayor que la de obtener un 2, siendo máxima la probabilidad de obtener un 7. De ahí que éste sea el número que elijan los jugadores con mayor frecuencia. Al jugar al póker, la probabilidad de obtener un trío es inferior a la de obtener una pareja. Por ello, se valora menos esta última jugada.

Todo esto se debe al hecho de que el número de casos favorables no es el mismo en todos los casos. Así, el número de casos favorables de obtener una suma de 2 al lanzar dos dados es 1 (ambos dados con un 1), mientras que es de 2 en el caso de una suma igual a 3

	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	10
5	6	7	8	9	10	11
6	7	8	9	10	11	12

Fig. 4. Posibles sumas de dos dados.

SUMA	CASOS FAVORABLES	TOTAL
2	{1,1}	1
3	{1,2}, {2,1}	2
4	{1,3}, {2,2}, {3,1}	3
5	{1,4}, {2,3}, {3,2}, {4,1}	4
6	{1,5}, {2,4}, {3,3}, {4,2}, {5,1}	5
7	{1,6}, {2,5}, {3,4}, {4,3}, {5,2}, {6,1}	6
8	{2,6}, {3,5}, {4,4}, {5,3}, {6,2}	5
9	{3,6}, {4,5}, {5,4}, {6,3}	4
10	{4,6}, {5,5}, {6,4}	3
11	{5,6}, {6,5}	2
12	{6,6}	1

Fig. 5. Casos favorables al lanzar dos dados.

(el primer dado con un 1 y el segundo con un 2, o el primero con un 2 y el segundo con un 1).

Hemos construido un programa con la misma filosofía de los anteriores. El lanzamiento de los dos dados se simula con dos variables I y J, que guardan números aleatorios entre 1 y 6. Utilizamos una variable S para la suma de los dos dados. El contador C(S) acumula los resultados obtenidos. Si el número de experimentos realizados es grande, los resultados obtenidos tenderán en cada caso a su probabilidad, como se refleja en la figura 6.

SUMA	PROBABILIDAD
2	1/36 = 0.028
3	2/36 = 0.056
4	3/36 = 0.083
5	4/36 = 0.111
6	5/36 = 0.139
7	6/36 = 0.167
8	5/36 = 0.139
9	4/36 = 0.111
10	3/36 = 0.083
11	2/36 = 0.056
12	1/36 = 0.028

Fig. 6. Cálculo de probabilidades.

```

10 REM *****
20 REM * CALCULO DE FRECUENCIAS *
30 REM * AL TIRAR DOS DADOS *
40 REM *****
50 REM
60 REM *****
70 REM * IBM,MSX,SPECTRUM,AMSTRAD,COMMODORE *
80 REM *****
90 CLS
100 DIM C(12):LET C=0
110 INPUT "NUMERO DE TIRADAS";N
120 CLS
130 PRINT " ESPERA UN MOMENTO "
140 REM *****
150 REM * TIRADAS DE LOS DADOS *
160 REM *****
170 FOR A=1 TO N
180 RANDOMIZE TIMER
190 LET I=INT(RND*6)+1
200 LET J=INT(RND*6)+1
210 LET S=I+J
220 LET C(S)=C(S)+1
230 LET C=C+1
240 NEXT A
250 CLS
260 REM *****
270 REM * RESULTADOS *
280 REM *****
290 PRINT "NUMERO DE TIRADAS:";N
300 PRINT:PRINT
310 PRINT "NUMERO","PORCENTAJE"
320 FOR S=2 TO 12
330 PRINT S,C(S);"/";N
340 NEXT S
350 PRINT:PRINT
360 PRINT "TOTAL:";C;"/";N

```

Programa 3.

Modificaciones para otros equipos

SPECTRUM

180 RANDOMIZE 0

AMSTRAD

180 RANDOMIZE TIMER

MSX y COMMODORE

No poner la línea 180.

190 I=INT(RND(1)*6)+1

200 J=INT(RND(1)*6)+1

Además, en el Commodore, sustituir las instrucciones CLS por PRINT CHR\$(143)

En este tipo de fenómenos es muy interesante la representación gráfica mediante un diagrama de barras. En este caso hemos utilizado barras verticales para representar gráficamente el lanzamiento de dos dados. Hemos construido un programa que habría que añadir al programa anterior. El programa se divide en cuatro partes:

1. Cálculo de la máxima frecuencia.
2. Transformación a escala de todas las frecuencias.
3. Representación gráfica en forma de diagrama de barras verticales.
4. Representación gráfica adjunta del contorno de las frecuencias.

Para calcular el valor máximo de las frecuencias se utiliza el método de la variable de intercambio. Se nombra una variable como MAXIMO y se le asigna uno de los valores del conjunto de frecuencias. Posteriormente se compara esta variable con el resto del conjunto. Si uno de los valores es mayor que el máximo, se le asigna a la variable MAXIMO dicho valor, y así sucesivamente.

Una vez calculado el máximo, se transforman todos los valores de la frecuencia multiplicándolos por 15 y dividiéndolos por el máximo, para que "quepan" en la pantalla.

Posteriormente se representan los valores transformados de las frecuencias como barras verticales. Para ello se convierte cada valor en un número igual de cuadrados negros. Con este objetivo se utiliza un bucle variable que llegue hasta el valor transformado de la frecuencia. Cada cuadrado se sitúa en el lugar adecuado mediante una instrucción LOCATE o PRINT AT.

APRENDER CON EL ORDENADOR

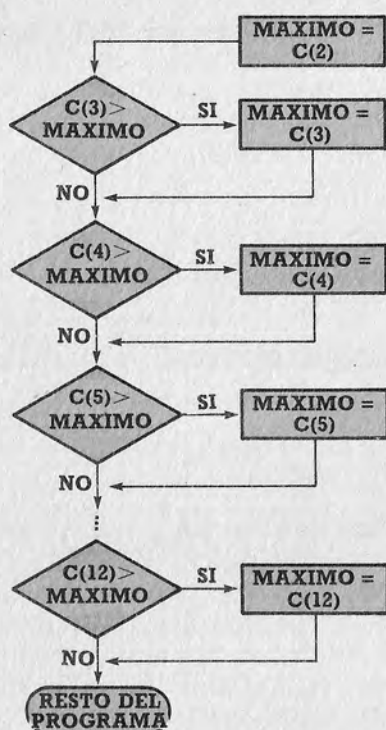
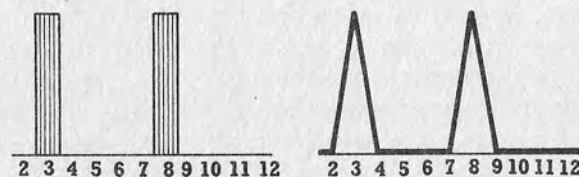
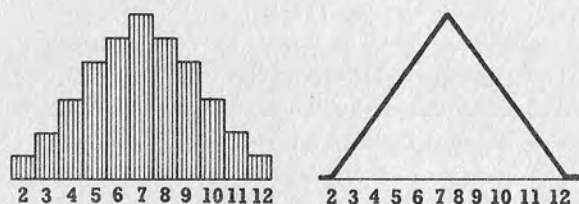


Fig. 7. Organigrama para el cálculo del máximo.

Por último, se realiza una representación gráfica del contorno de las frecuencias. Si el número de experimentos es muy grande el contorno tiende a ser un triángulo.



N.º EXPERIMENTOS: 2



N.º EXPERIMENTOS: 1.000

Fig. 8. Representación gráfica de las frecuencias obtenidas al lanzar dos dados.

```

500 REM *****
510 REM * REPRESENTACION GRAFICA *
520 REM * DEL PROGRAMA DE CALCULO DE *
530 REM * FRECUENCIAS AL TIRAR DOS DADOS *
540 REM *****
550 REM
560 REM *****
570 REM * UNIR AL PROGRAMA ANTERIOR *
580 REM *****
590 REM
600 REM *****
610 REM * VALIDO PARA IBM *
620 REM *****
630 INPUT "PULSA RETORNO PARA CONTINUAR",A$
640 CLS
650 SCREEN 2
660 DIM G(13)
670 REM *****
680 REM * CALCULO DEL MAXIMO *
690 REM *****
700 LET MAXIMO=C(2)
710 FOR S=3 TO 12
720 IF C(S)>MAXIMO THEN LET MAXIMO=C(S)
730 NEXT S
740 REM *****
750 REM * TRANSFORMACION A ESCALA *
760 REM *****
770 FOR S=2 TO 12
780 LET G(S)=C(S)*15/MAXIMO
790 NEXT S
800 REM *****
810 REM * DIAGRAMA DE BARRAS VERTICALES *
820 REM *****
830 FOR S=2 TO 12
840 FOR T=1 TO G(S)
850 LOCATE 21-T,S+10:PRINT CHR$(143)
860 NEXT T
870 NEXT S
880 REM *****
890 REM * REPRESENTACION DEL CONTORNO *
900 REM *****
910 FOR S=2 TO 13
920 LINE (200+20*S+10,160-G(S)*8)-(200+20*(S-1)+10,160-G(S-1)*8)
930 NEXT S
  
```

Programa 4.

■ Modificaciones para otros equipos

En todos los casos eliminar la línea 650.
En el MSX sustituirla por SCREEN 1.

SPECTRUM

```
850 PRINT AT 21-T,S+10;CHR$(143)
```

Ajustar la línea 920 al formato de DRAW.

AMSTRAD

```
850 LOCATE S+10,21-T:PRINT CHR$(143)
```

Ajustar la línea 920 al formato de DRAW.

SOCIEDAD

■ Geografía: Capitales del mundo

En esta sección presentamos un programa de tipo test sobre las capitales mundiales. En el programa se utiliza una variable C\$(I,J,K) que depende de tres subíndices:

- I: para el continente.
- J: para el país.
- K: para la ciudad.

Cada país tiene cuatro datos. El primero es el nombre del país, el segundo es su ca-

pital. El tercero y el cuarto corresponden a las dos ciudades más importantes.

C\$(I,J,1): NOMBRE DEL PAIS

C\$(I,J,2): CAPITAL

C\$(I,J,3): CIUDAD MAS IMPORTANTE

C\$(I,J,4): SEGUNDA CIUDAD MAS IMPORTANTE

Con el objeto de no restringir el programa a un número concreto de datos, hemos empleado el "truco" de incluir un dato "marcador" para determinar cuándo se terminan los datos de un continente. Un contador C(I) se encarga de calcular el número de países por continente.

En el programa se presenta un menú de opciones para elegir el continente. Después se extrae aleatoriamente una variable que asignará el país concreto sobre el que se va a preguntar.

En cada continente el número de preguntas es cinco.

En todos los casos, para evitar que se conozca "a priori" cuál es la capital, se ordenan alfabéticamente las alternativas. Estas no son otras que las tres ciudades de cada país. Para ello se emplea el método de indexación consistente en asignar un número de orden a cada función, según sea su posición alfabética.

```
10 REM *****
20 REM * CAPITALES MUNDIALES *
30 REM *****
40 REM
50 REM *****
60 REM * VALIDO PARA *
70 REM * IBM,AMSTRAD,MSX,COMMODORE *
80 REM *****
90 CLS
100 REM *****
110 REM * INICIALIZACION DE VARIABLES *
120 REM *****
130 DIM C$(4,20,4):DIM C(4):DIM R(20):DIM O(4)
140 LET BIEN=0:LET MAL=0
150 REM *****
160 REM * ENTRADA DE DATOS *
170 REM *****
180 FOR I=1 TO 4
190 FOR J=1 TO 20
200 FOR K=1 TO 4
210 READ C$(I,J,K)
220 NEXT K
230 IF C$(I,J,1)="FIN" THEN GOTO 260
240 LET C(I)=C(I)+1
250 NEXT J
260 NEXT I
270 CLS
280 REM *****
290 REM * MENU DE OPCIONES *
300 REM *****
310 PRINT "CAPITALES MUNDIALES"
320 PRINT "1.EUROPA"
330 PRINT "2.AMERICA"
340 PRINT "3.ASIA"
```

APRENDER CON EL ORDENADOR

```

350 PRINT "4.AFRICA"
360 PRINT "5.FINALIZAR CON EL PROGRAMA"
370 PRINT:PRINT
380 INPUT "ELIGE CONTINENTE (1 a 4)";I
390 IF I=5 THEN END
400 REM *****
410 REM * EXTRACCION DE PREGUNTAS *
420 REM *****
430 GOSUB 5000
440 LET PREG=0
450 CLS
460 IF PREG=5 THEN GOTO 270
470 LET PREG=PREG+1
480 RANDOMIZE TIMER
490 LET J=INT(RND*C(I))+1
500 LET R(J)=R(J)+1
510 IF R(J)>1 THEN GOTO 490
520 PRINT "CUAL ES LA CAPITAL DE :";C$(I,J,1)
530 PRINT:PRINT
540 GOSUB 6000
550 GOSUB 1000
560 GOSUB 2000
570 REM *****
580 REM * COMPROBACION DE ACIERTOS *
590 REM *****
600 INPUT "CUAL ES LA CORRECTA";B
610 IF B=O(2) THEN PRINT "CORRECTO";LET BIEN=BIEN+1
620 IF B<>O(2) THEN PRINT "INCORRECTO, ERA:";C$(I,J,2);LET MAL=MAL+1
630 GOSUB 4000
640 FOR Z=1 TO 2000:NEXT Z
650 GOTO 450
1000 REM *****
1010 REM * PROCESO DE ORDENACION POR INDEXACION *
1020 REM *****
1030 FOR K=2 TO 4
1040 FOR L=2 TO 4
1050 IF C$(I,J,K)>C$(I,J,L) THEN LET O(K)=O(K)+1
1060 NEXT L:NEXT K
1070 RETURN
2000 REM *****
2010 REM * BUSQUEDA DE OPCIONES *
2020 REM *****
2030 FOR K=1 TO 3
2040 FOR L=2 TO 4
2050 IF O(L)=K THEN GOSUB 3000
2060 NEXT L:NEXT K
2070 RETURN
3000 REM *****
3010 REM * VISUALIZACION DE OPCIONES *
3020 REM *****
3030 PRINT K;"-";C$(I,J,L)
3040 RETURN
4000 REM *****
4010 REM * MARCADOR DE LOS RESULTADOS *
4020 REM *****
4030 PRINT:PRINT
4040 PRINT:PRINT "CORRECTAS:";BIEN
4050 PRINT:PRINT "INCORRECTAS:";MAL
4060 RETURN
5000 REM *****
5010 REM * INICIALIZACION DEL CONTADOR DE CAPITALES PREGUNTADAS *
5020 REM *****
5030 FOR J=1 TO C(I)
5040 LET R(J)=0
5050 NEXT J
5060 RETURN
6000 REM *****
6010 REM * INICIALIZADOR DEL CONTADOR DE INDICES *
6020 REM *****
6030 FOR K=2 TO 4
6040 LET O(K)=1
6050 NEXT K
6060 RETURN
7000 REM *****
7010 REM ** DATOS **
7020 REM *****
7030 REM
7040 REM *****
7050 REM ** CAPITALES DE **
7060 REM ** EUROPA **
7070 REM *****
7080 DATA "ALBANIA","TIRANA","DURAZZO","SCUTARI"

```

```

7090 DATA "BELGICA", "BRUSELAS", "AMBERES", "GANTE"
7100 DATA "ESPAÑA", "MADRID", "BARCELONA", "VALENCIA"
7110 DATA "FRANCIA", "PARIS", "MARSELLA", "LYON"
7120 DATA "ITALIA", "ROMA", "MILAN", "NAPOLIS"
7130 DATA "SUECIA", "ESTOCOLMO", "GOTEBORG", "MALMOE"
7140 DATA "URSS", "MOSCU", "LENINGRADO", "KIEV"
7150 DATA "YUGOSLAVIA", "BELGRADO", "OSIJEK", "ZAGREB"
7160 DATA "REINO UNIDO", "LONDRES", "BIRMINGHAM", "GLASGOW"
7170 DATA "BULGARIA", "SOFIA", "PLOVDIV", "VARNA"
7180 DATA "CHECOSLOVAQUIA", "PRAGA", "BRATISLAVA", "BRNO"
7190 DATA "FINLANDIA", "HELSINKI", "TAMPERE", "TURKU"
7200 DATA "FIN", "FIN", "FIN", "FIN"
7210 DATA "MEJICO", "MEXICO", "GUADALAJARA", "MONTERREY"
7220 REM *****
7230 REM ** CAPITALES DE **
7240 REM ** AMERICA **
7250 REM *****
7260 DATA "ARGENTINA", "BUENOS AIRES", "CORDOBA", "ROSARIO"
7270 DATA "BRASIL", "BRASILIA", "SAO PAULO", "RIO DE JANEIRO"
7280 DATA "CHILE", "SANTIAGO", "CONCEPCION", "VALPARAISO"
7290 DATA "COLOMBIA", "BOGOTA", "MEDELLIN", "CALI"
7300 DATA "ESTADOS UNIDOS", "WASHINGTON", "NUEVA YORK", "CHICAGO"
7310 DATA "CANADA", "OTAWA", "TORONTO", "MONTREAL"
7320 DATA "FIN", "FIN", "FIN", "FIN"
7330 REM *****
7340 REM ** CAPITALES DE **
7350 REM ** ASIA **
7360 REM *****
7370 DATA "AFGANISTAN", "CABUL", "KANDAKHAR", "HERAT"
7380 DATA "BANGLADESH", "DACCÁ", "CHITTAGUNG", "KHULNA"
7390 DATA "BIRMANIA", "RANGUN", "MANDALAY", "MULMEIN"
7400 DATA "FILIPINAS", "MANILA", "QUEZON", "DAVAO"
7410 DATA "INDIA", "NUEVA DELHI", "CALCUTA", "BOMBAY"
7420 DATA "INDONESIA", "YAKARTA", "SURABAYA", "BANDUNG"
7430 DATA "JAPON", "TOKIO", "YOKOHAMA", "OSAKA"
7440 DATA "JORDANIA", "AMMAN", "ZARQA", "IRBID"
7450 DATA "FIN", "FIN", "FIN", "FIN"
7460 REM *****
7470 REM ** CAPITALES DE **
7480 REM ** AFRICA **
7490 REM *****
7500 DATA "ARGELIA", "ARGEL", "ORAN", "CONSTANTINA"
7510 DATA "CAMERUN", "YAUNDE", "DOUALA", "MAROUA"
7520 DATA "EGIPTO", "EL CAIRO", "ALEJANDRIA", "GIZA"
7530 DATA "GHANA", "ACCRA", "KUMASI", "TAMALE"
7540 DATA "LIBIA", "TRIPOLI", "BENGAZI", "MISURATA"
7550 DATA "MALI", "BAMAKO", "SEGOU", "MOPTI"
7560 DATA "MARRUECOS", "RABAT", "CASABLANCA", "MARRAKECH"
7570 DATA "SENEGAL", "DAKAR", "THIES", "KAOLACK"
7580 DATA "SUDAN", "JARTUM", "OMDURMAN", "PORT SUDAN"
7590 DATA "UGANDA", "KAMPALA", "JINJA", "MASAKA"
7600 DATA "ZAIRE", "KINSHASA", "KANANGA", "LUMUMBASHI"
7610 DATA "FIN", "FIN", "FIN", "FIN"

```

Programa 5.

■ Modificaciones para otros equipos

INSTRUCCION CLS

Sustituirla siempre en el Commodore por PRINT CHR\$(143)

INSTRUCCION RANDOMIZE TIMER

Suprimirla en Commodore y MSX.

Sustituirla en Amstrad por RANDOMIZE TIME y en el Spectrum por RANDOMIZE 0.

INSTRUCCION END

Sustituirla en el Spectrum por GOTO 9999.

INSTRUCCION RND

Sustituirla en Commodore y MSX por RND(1).

DIM C\$(4,20,4)

Añadir un cuarto subíndice para el Spectrum.

PARA LOS MAS JOVENES

■ Geografía: Comunidades Autónomas

Hasta hace unos pocos años España estaba dividida en regiones históricas. En algunos casos, la división no era del todo adecuada, por lo que se procedió a establecer una serie de Comunidades Autónomas, más de acuerdo con las necesidades actuales.

En esta sección hemos diseñado un programa que permitirá conocer esta nueva divi-

APRENDER CON EL ORDENADOR

sión. Para ello se realizan diez preguntas de tipo test en el que se debe adivinar a qué Comunidad Autónoma pertenece una provincia cualquiera.

Los datos se guardan en una matriz de

dos subíndices, C\$(I,J). El primer subíndice está relacionado con la Comunidad Autónoma, mientras que el segundo hace referencia a cada provincia. Como el número de provincias de cada Comunidad es variable, se utiliza un contador C(I) para determinarlo.

```
10 REM *****
20 REM * PROGRAMA DE LAS COMUNIDADES AUTONOMAS *
30 REM * ESPANOLAS *
40 REM *****
50 REM
60 REM *****
70 REM * VALIDO PARA *
80 REM * IBM,AMSTRAD,MSX,SPECTRUM,COMMODORE *
90 REM *****
100 CLS
110 REM *****
120 REM * INICIALIZACION DE VARIABLES *
130 REM *****
140 DIM C$(17,11):DIM C(17):DIM P(4):DIM R(17):DIM O(17)
150 REM *****
160 REM * LECTURA DE DATOS *
170 REM *****
180 FOR I=1 TO 17
190 FOR J=1 TO 11
200 READ C$(I,J)
210 IF C$(I,J)="FIN" THEN GOTO 240
220 LET C(I)=C(I)+1
230 NEXT J
240 NEXT I
250 REM *****
260 REM * EXTRACCION DE PREGUNTAS *
270 REM *****
280 GOSUB 4000
290 RANDOMIZE TIMER
300 LET P(1)=INT(RND*17)+1
310 LET R(P(1))=R(P(1))+1
320 LET K=INT(RND*(C(P(1))-1))+2
330 FOR I=2 TO 4
340 LET P(I)=INT(RND*17)+1
350 LET R(P(I))=R(P(I))+1
360 IF R(P(I))>1 THEN GOTO 340
370 NEXT I
380 FOR I=1 TO 4
390 FOR J=1 TO 4
400 IF C$(P(I),J)>C$(P(J),1) THEN LET O(I)=O(I)+1
410 NEXT J:NEXT I
420 CLS
430 REM *****
440 REM * VISUALIZACION DE PREGUNTAS *
450 REM *****
460 PRINT "LA PROVINCIA DE:";C$(P(1),K)
470 PRINT "A QUE COMUNIDAD AUTONOMA PERTENECE?"
480 PRINT:PRINT
490 REM *****
500 REM * ORDENACION DE ALTERNATIVAS *
510 REM *****
520 FOR I=1 TO 4
530 FOR J=1 TO 4
540 IF O(J)=I THEN PRINT I;"-";C$(P(J),1)
550 NEXT J
560 NEXT I
570 REM *****
580 REM * PREGUNTA *
590 REM * Y COMPROBACION DE LA RESPUESTA *
600 REM *****
610 INPUT "CUAL ES LA CORRECTA";B
620 IF B=O(1) THEN GOSUB 1000
630 IF B<>O(1) THEN GOSUB 2000
640 LET PR=PR+1
650 FOR Z=1 TO 1000:NEXT Z
660 IF PR<10 THEN GOTO 280
670 END
1000 REM *****
1010 REM * RESPUESTA CORRECTA *
1020 REM *****
```

```

1030 PRINT "C O R R E C T O"
1040 LET BIEN=BIEN+1
1050 GOSUB 3000
1060 RETURN
2000 REM *****
2010 REM * RESPUESTA INCORRECTA *
2020 REM *****
2030 PRINT "I N C O R R E C T O"
2040 PRINT:PRINT "ERA LA COMUNIDAD DE:";C$(P(1),1)
2050 LET MAL=MAL+1
2060 GOSUB 3000
2070 RETURN
3000 REM *****
3010 REM * MARCADOR *
3020 REM *****
3030 PRINT:PRINT
3040 PRINT "CORRECTAS:";BIEN
3050 PRINT "INCORRECTAS:";MAL
3060 RETURN
4000 REM *****
4010 REM * INICIALIZACION DE CONTADORES *
4020 REM *****
4030 FOR I=1 TO 17
4040 LET O(I)=1
4050 LET R(I)=0
4060 NEXT I
4070 RETURN
6000 REM *****
6010 REM * DATOS *
6020 REM *****
6030 DATA "ANDALUCIA","ALMERIA","CADIZ","CORDOBA","GRANADA"
6040 DATA "HUELVA","JAEN","MALAGA","SEVILLA","FIN"
6050 DATA "ARAGON","HUESCA","TERUEL","ZARAGOZA","FIN"
6060 DATA "ASTURIAS","ASTURIAS","FIN"
6070 DATA "BALEARES","BALEARES","FIN"
6080 DATA "CANARIAS","LAS PALMAS","SANTA CRUZ DE TENERIFE","FIN"
6090 DATA "CANTABRIA","CANTABRIA","FIN"
6100 DATA "CASTILLA-LA MANCHA","ALBACETE","CIUDAD REAL","CUENCA"
6110 DATA "GUADALAJARA","TOLEDO","FIN"
6120 DATA "CASTILLA-LEON","AVILA","BURGOS","LEON","PALENCIA","SALAMANCA"
6130 DATA "SEGOVIA","SORIA","VALLADOLID","ZAMORA","FIN"
6140 DATA "CATALUNA","BARCELONA","GERONA","LERIDA","TARRAGONA","FIN"
6150 DATA "EXTREMADURA","BADAJOZ","CACERES","FIN"
6160 DATA "GALICIA","LA CORUNA","LUGO","ORENSE","PONTEVEDRA","FIN"
6170 DATA "LA RIOJA","LA RIOJA","FIN"
6180 DATA "MADRID","MADRID","FIN"
6190 DATA "MURCIA","MURCIA","FIN"
6200 DATA "NAVARRA","NAVARRA","FIN"
6210 DATA "PAIS VASCO","ALAVA","GUIPUZCOA","VIZCAYA","FIN"
6220 DATA "COMUNIDAD VALENCIANA","ALICANTE","CASTELLON","VALENCIA","FIN"

```

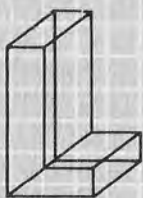
Programa 6.

■ Modificaciones para otros equipos

Ver indicaciones dadas en el programa anterior.

PEQUEÑA HISTORIA DE LA INFORMATICA

■ La generación de ordenadores (II)



A tercera generación de ordenadores supuso otro cambio trascendental. Es verdad que durante la segunda, los ordenadores se habían ido perfeccionando poco a poco, habían aparecido nuevos sistemas de almacenamiento, mejores memorias, nuevos lenguajes de programación de nivel superior, y adaptados a distintas necesidades, pero el acontecimiento trascendental que marca esta época es la aparición del circuito integrado o chip.

Fue un invento revolucionario. Un chip o circuito integrado es un pequeño rectángulo, de un tamaño parecido al de un transistor, pero en el que se encuentran unidos muchos componentes (hoy en día son decenas de miles). Este proceso de miniaturización contribuyó mucho al abaratamiento de los equipos, a aumentar su fiabilidad y su velocidad (los componentes estaban muy próximos unos a otros, de forma que la velocidad de transferencia de los datos aumentó de forma espectacular).

Como antes hemos indicado, también decreció la tasa media de averías, funcionando las máquinas miles de horas sin que se produjeran problemas. Pero sin duda, la velocidad fue la característica que hizo obsoletas a las máquinas anteriores de la segunda generación.

La velocidad de cálculo pasó de unos milisegundos a varios nanosegundos (10 elevado a menos tres a 10 elevado a menos seis).

Naturalmente, al procesarse las instrucciones con tan enorme rapidez, pasó a tener una importancia grande el que se pudieran ejecutar varios programas a la vez (naturalmente, los datos siempre se procesan de forma secuencial, pero mientras se va procesando el programa en la unidad central, las otras unidades y partes de la máquina se usan en otras tareas). Sin embargo, ya no es necesario disponer de un ordenador auxiliar para las tareas de selección y prioridades. A este sistema de procesamiento se le denomina "multiprogramación". Al eliminar el segundo ordenador, la memoria del ordenador principal se reconfigura, o se "divide" en dos zonas, una intermedia y otra de procesamiento. En una de estas zonas se almacena la "cola" para su proceso paulatino, y en otra, el programa o datos que se van a procesar en ese momento. Otra mejora a este sistema es la introducción de procesos desde terminales (teleproceso).

La velocidad es una característica muy importante en un ordenador. Ciertos cálculos no podían llevarse a cabo si la velocidad de la máquina no alcanza determinados niveles. La razón es que ciertos cálculos muy complicados o largos tomarían un tiempo de proceso tan largo que sería impracticable (por ejemplo, hay casos en los que es necesario considerar muchos factores y realizar los cálculos para cada uno de ellos, sacando a continuación las conclusiones oportunas). Al aumentar la velocidad aumentó la potencia del ordenador, pudiendo realizar cálculos que antes eran impensables (pensemos en la predicción del tiempo, en Meteorología).

Dentro del proceso de integración, no todos los ordenadores estaban integrados en idéntico grado. Así, en los libros, aparece una

división de las máquinas atendiendo a su grado de integración (VLSI, integración a muy gran escala; SSI, a pequeña escala; MSI, a media escala; LSI, a gran escala, etc.).

El problema que surgió a continuación en esta enorme revolución fue la compatibilidad. Las nuevas máquinas no eran en absoluto compatibles con las de la generación anterior, que habían quedado totalmente obsoletas. Las conversiones resultaron muy caras y complejas. Sin embargo, las casas fabricantes eliminaron los problemas de compatibilidad entre las máquinas que ofrecían, simplificando totalmente los problemas de conversión entre un ordenador y otro más potente de la misma casa.

En estos años se desarrollaron también mucho los sistemas de almacenamiento permanente, muy importantes para aumentar la capacidad 'real' de memoria. Aparecieron las memorias de disco y otros tipos de memoria muy mejorados. El acceso a este tipo de memoria es bastante más lento, pero, de todos modos, flexibiliza enormemente el proceso.

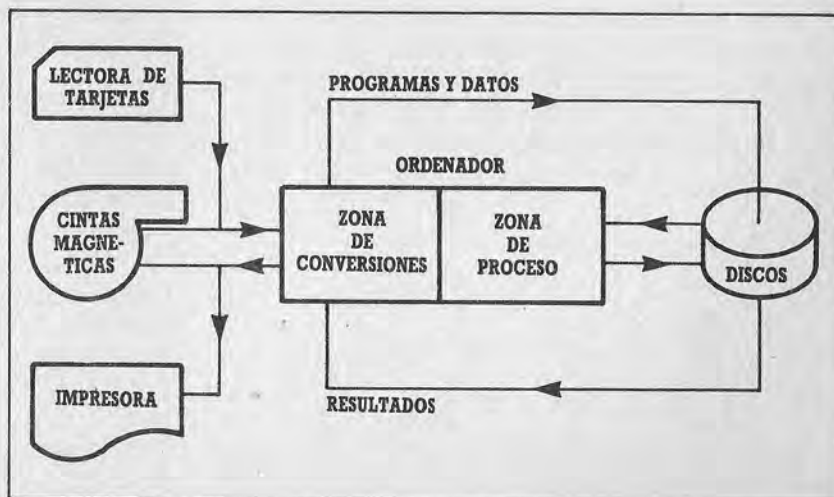
En 1964 IBM presenta la serie 360 (circuitos integrados) para sustituir a la serie 7000. En esta serie introduce la memoria virtual, es decir, un sistema de acceso a la memoria de almacenamiento masivo, como si fuera la memoria central. El nombre virtual viene de que actúa virtualmente como si fuera la memoria central, en lo que respecta a acceder a ella, y la única salvedad es que el tiempo requerido es algo mayor.

La capacidad de memoria de un ordenador se medía en bits, es decir, dígitos binarios, y en palabras. Normalmente se consideraba que las palabras tenían treinta y dos bits.

Más tarde aparece otra forma de designar la memoria, midiéndola en bytes o grupos de ocho bits. Las palabras pueden tener dos o cuatro bytes. Algunos ordenadores procesan por palabras de 16 bits, entre ellos la serie PDP de Digital.

Si consideramos la demanda de ordenadores de la época, veremos que era tremendamente cambiante, y que iban apareciendo multitud de firmas creadoras de ordenadores muy especializados (adaptados a las distintas necesidades). Así aparecen los "miniordenadores". Son ordenadores muy pequeños que pueden unirse unos a otros para llevar a cabo distintas tareas (no pensando en el usuario doméstico). Los padres de la idea son los fundadores de Digital Equipment Corporation, Harlan Anderson y Stan y Ken Olsen. Desde 1957 produjeron este tipo de ordenadores, mejorándolos poco a poco. En el año 1959 apareció el PDP-1, que revolucionó el mercado de ordenadores, por su precio y versatilidad. Se vendió a escuelas, empresas, laboratorios de todo tipo. Además, la política comercial de la empresa era ambiciosa, y pensando en el futuro, regalaron ordenadores a muchas universidades, considerando que los estudiantes, conocedores de la máquina y acostumbrados a ella, serían en el futuro hombres de empresa y posibles clientes.

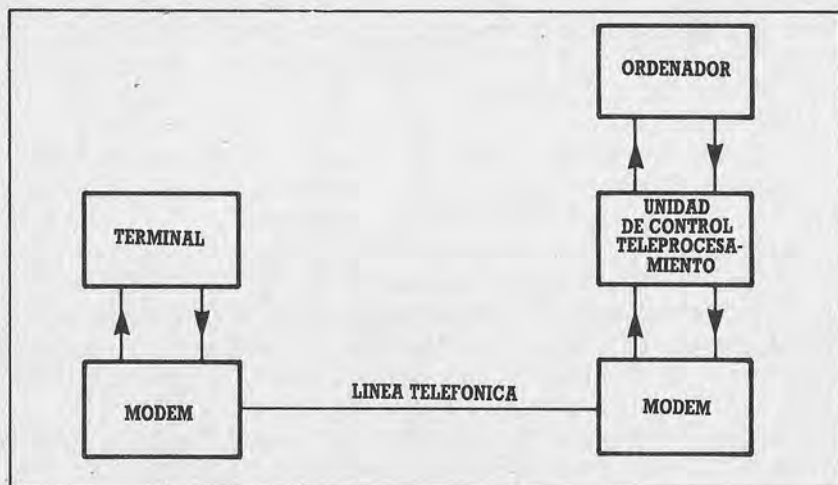
La cuarta generación de ordenadores supone también grandes cambios, pero éstos se van produciendo de forma más gradual que en las generaciones anteriores. Una de sus características principales es la integración a gran escala. Además, también cambia el sistema de explotación de los grandes equipos, y sobre todo, aparece el microordenador u or-



Los ordenadores de la tercera generación son capaces de ejecutar varios programas simultáneamente. Para ello la memoria del ordenador está dividida en dos zonas: de "conversiones" y zona de "proceso".

PEQUEÑA HISTORIA DE LA INFORMATICA

El teleprocesamiento permite al usuario de sistemas informáticos introducir los datos y recibir los resultados de un ordenador situado en cualquier otro lugar, siempre que exista la comunicación telefónica.



denador doméstico. El auge de estas pequeñas máquinas y sus múltiples aplicaciones revolucionan la vida cotidiana de nuestro planeta.

El microprocesador aparece en 1971, creado en los laboratorios de desarrollo de la Texas Instruments. Es un medio de integración a muy gran escala. Consiste en una pequeña oblea de silicio, en la que se encuentra miniaturizado todo un circuito electrónico. Su precio es relativamente barato, y la gama de sus posibilidades enorme, ya que se puede utilizar para microordenadores y también en todo tipo de sistemas de automatización (discos de tráfico, máquinas de juego, ascensores, etc.). Además, estos microprocesadores podían cablearse y pasar a realizar funciones específicas, pudiendo pasar de una a otra función a través de los programas de software.

Intel también fabricó plaquetas para la empresa japonesa Busicom. Unió once en una misma, programable, la 4004. Como todo chip, al no tener conexiones, era muy fiable.

Los lenguajes de programación sufren un desarrollo enorme, mejorándose los ya surgidos anteriormente, y apareciendo otros nuevos. Por ejemplo, BASIC, creado en Dartmouth (EE.UU.), fue muy mejorado, y utilizado en la mayoría de las pequeñas máquinas. Es un lenguaje de muy alto nivel, y relativamente fácil

de aprender, lo que acerca el microordenador al público, haciendo interesarse por estos temas a personas que no tenían una necesidad fuerte de disponer de un ordenador.

Realmente, lo único que no ha bajado

¿Sabía usted que...

¿Se ha parado usted a pensar cómo pueden llevar a cabo los bancos tantas operaciones entre distintas entidades?

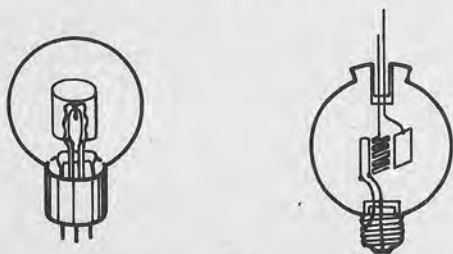
Cuando usted ingresa un talón, en el banco le hacen esperar tres días para hacerlo efectivo. Pero ¿cómo se realiza la operación?

Existe un organismo llamado Cámara de Compensación Automática. Este nombre indica realmente un centro de proceso de datos muy sofisticado. En él, la velocidad y seguridad de la información son absolutamente esenciales.

¿Qué información se procesa? Las cintas que aporta cada banco o entidad. Así, a una velocidad enorme, los datos (las distintas órdenes) pasan de una cinta a otra, efectuándose los distintos pagos, ingresos, etc., sin que aparezcan billetes ni cheques.

Pero tampoco es necesario enviar físicamente las cintas para su proceso posterior. En efecto, los datos pueden enviarse por algún sistema de comunicación suficientemente fiable, que evite la acción física de llevar la cinta.

Existen muchas empresas internacionales que utilizan estos recursos para facilitar sus operaciones bancarias. Por ejemplo, las tarjetas de compras. Con alguna tarjeta de compras internacional usted puede comprar en Venecia, sacar dinero, etc., y el importe le será descontado de su cuenta en un tiempo muy reducido.



Esquema de las primeras válvulas.

de precio desde la aparición del primer ordenador es el software. Naturalmente, esto se debe a que la creación de software es una tarea intelectual de seres humanos, y su precio se va alterando, al contrario que el de las máquinas. Mientras éstas bajan, los salarios suben, y el precio de las aplicaciones se ve afectado por ello. De todos modos, y aunque existen innumerables posibilidades de aplicaciones a medida, en el mercado se encuentran paquetes muy útiles para multitud de aplicaciones.

En 1965 Andrew Bobeck diseña un nuevo modelo de memoria: la memoria de burbujas, en los laboratorios de la Bell. Este tipo de memoria es muy fiable, de bajo coste, puede almacenar datos a largo plazo y en general tiene más ventajas que las memorias de núcleo de ferrita. (Como consideramos que merece un estudio más extenso, hablaremos de ella en otro capítulo.) Aparecen las memorias de acceso aleatorio (RAM), frente a las memorias de lectura (ROM). Intel creó también una microplaqueta de memoria RAM de 256 K, que ayudó también a simplificar la estructura de los microordenadores.

También mejoran considerablemente los equipos periféricos. La plaqueta 8008 de Intel controla la pantalla de visualización de rayos catódicos. Procesaba ocho líneas en paralelo, frente a 4, que era lo normal. Intel no tenía demasiada confianza, pero el 8008 se vendió muchísimo.

También fabricaron microchips Motorola (6800), Fairchild, NS, y otras firmas hacia la mitad de los años setenta.

En esa misma época, la empresa MIT, dedicada no específicamente a ordenadores, estudia el mercado y decide ofrecer al gran

público un ordenador, introduciendo el microprocesador en pequeñas máquinas, muy versátiles y baratas. El resultado es el Altair 8800, que llevaba un microprocesador 8080 de Intel.

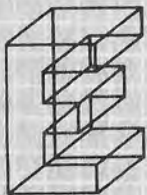
Más adelante, a estos microordenadores se les han ido añadiendo sistemas de almacenamiento sofisticados y potentes, como discos duros, también discos flexibles, impresoras, sistemas de visualización avanzados, con gran resolución, etc.

Pero aunque la cuarta generación se distingue fundamentalmente por la aparición de los pequeños microordenadores domésticos, eso no significa en absoluto que no existan otro tipo de ordenadores enormes, muy especializados, en los que prima fundamentalmente la velocidad de proceso, y, por tanto, la capacidad de procesar enormes cantidades de datos (el ordenador Cray-1 es uno de ellos).

Estos ordenadores tienen una arquitectura diferente y utilizan otro tipo de tecnología. En este caso, el tamaño pasa a tener menos importancia, primándose la velocidad de cálculo y la memoria rápida y grande (estos ordenadores suelen utilizar transistores Mit).

El 3.800 de IBM aporta un sistema de almacenamiento masivo muy interesante. No sabemos si la idea surgió de alguno de los técnicos mientras daba un paseo por el campo, en primavera, disfrutando del olor de las flores, mientras bordoneaban junto a él las activas abejas (Newton también tuvo alguna idea interesante mientras paseaba entre los manzanos cargados de fruta madura). El sistema de almacenamiento masivo de IBM consiste en unas pequeñas celdillas hexagonales (para aprovechar integralmente el espacio), en cuyo interior se encuentran cartuchos con cintas magnéticas.

■ Los ordenadores de la quinta generación



NTRE los días 19 y 22 de octubre de 1981 tuvo lugar en Tokio la Conferencia Internacional de la Quinta Generación. En ella se intentaba llegar a dilucidar los planes a seguir internacionalmente para llevar a cabo la empresa de crear un ordenador para los años noventa. La Conferencia tuvo lugar en Tokio, porque los padres de la idea eran japoneses, aunque querían dar cabida en el proyecto a todas las naciones interesadas.

Realmente, todo lo que aconteció en dicha Conferencia había ido fraguándose durante varios años, y la Conferencia sólo fue el resultado de mucho trabajo y planificación anteriores. También la preparación de unas buenas relaciones entre los distintos países participantes exigió viajes y dedicación, ya que, al intentar intercambios de información muy valiosos, las comunicaciones podían hacerse tensas y desconfiadas.

Como sabe el lector, los japoneses empezaron algo más tarde que otros países el desarrollo de la Informática. Sin embargo, hoy en día tienen una posición prominente, con tecnología muy avanzada. Los mercados mundiales han evolucionado mucho, y la mayoría de los países industrializados tienen sus empresas (al menos las de cierto tamaño, ya no muy grande) mecanizadas. ¿Cuál será la demanda en los años venideros? Esa pregunta surgía en las instituciones gubernamentales niponas a finales de la década anterior.

El MITI (Ministerio de Comercio Inter-

nacional e Industria), junto con otros organismos de investigación de electrónica e informática, como el Laboratorio Electrotécnico, se habían interesado por el futuro de la industria informática, concretamente les preocupaba el decenio de 1990. Era importante encauzar la producción promoviendo las ayudas mecanizadas a la investigación, de forma que los resultados obtenidos en estos campos fueran interesantes desde el punto de vista práctico, y además, comerciales. El Ministerio comenzó su labor con la creación de un Centro para el Desarrollo de la Informática, y promoviendo, además, un Comité para la Investigación sobre la Quinta Generación.

Designaron como presidente del Comité a Tohru Moto-Oka, persona muy respetada en los ambientes científicos, y crearon varias secciones dedicadas a tareas específicas (estudio de las posibles arquitecturas, tecnología básica, estudio del tipo de máquina que pedirá el mercado de ordenadores).

La idea de anticiparse en la creación de un ordenador del futuro tuvo mucha aceptación en el Japón y hubo multitud de personas interesadas en colaborar de alguna forma en el proyecto. Sin embargo, a pesar de la Conferencia, y de los muchos contactos que se llevaron a cabo previamente en Alemania, Fran-

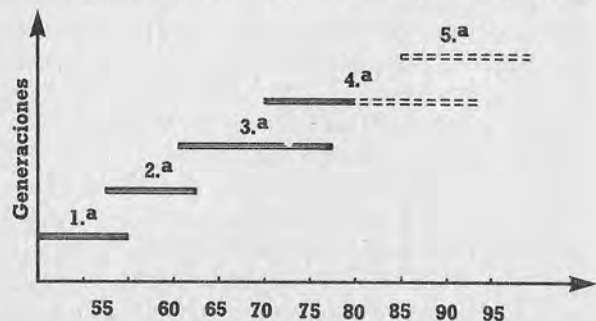


Fig. 1. Evolución de las generaciones de ordenadores.

cia, Gran Bretaña y Estados Unidos, la idea no entusiasmó todo lo que esperaban los japoneses, e incluso surgieron problemas de desconfianza, hasta llegar a acusaciones de "espionaje industrial".

Entre los occidentales existían y existen muchos detractores, que consideran que todo el proyecto: los documentos de planificación, la conferencia, el presupuesto, absolutamente todo es un gigantesco montaje, organizado para promocionar a corto plazo sus productos actuales y futuros. Sin embargo, no parece serio que los japoneses, grandes planificadores, se tomen tanto trabajo por algo, que si no consiguen se volverá en su contra, desprestigiándolos.

Pero, en primer lugar, veamos en qué consiste un ordenador de la quinta generación.

Para intentar averiguar qué tipo de ordenadores serán los que pida el mercado de esos años, fijémonos en los ordenadores actuales, y veamos de qué adolecen y cuáles son sus ventajas.

1. Nada más fijarnos en un ordenador, observamos que su primer inconveniente está en que su manejo no es asequible para el público de la calle, que necesitará por lo menos tomar algún curso de informática, o estudiar en casa algún que otro libro, para poder obtener algo interesante de él.

¡Fuera teclas! El ordenador debe comprender el lenguaje natural, y ejecutar las órdenes que se le den de viva voz. Este problema ya está bastante estudiado, pero no es todo lo sencillo que puede parecer a primera vista. Desde hace mucho tiempo se han intentado realizar trabajos de traducción de informes, etcétera, de un idioma a otro. Los resultados siempre han sido muy precarios, ya que el ordenador traducía literalmente, y como puede intuirse, ese tipo de traducciones no son, en absoluto, útiles. Para obtener una traducción válida, el ordenador debe "comprender" el entorno en el que se mueve la persona que le está ordenando, debe "saber" a qué se está refiriendo cuando le pide una determinada tarea. Para que dos personas se "comprendan" es necesario que tengan cosas comunes, sobre las que hablar. Si no comparten ese "saber" no podrán nunca comprenderse. Por ejemplo, si yo digo que mi hijo habla bastante bien inglés porque ha ido al colegio alemán, nadie que no esté ligado en cierto modo a dicho colegio (por ser padre, amigo, etc.), me

comprenderá en absoluto. Los que me comprenden comparten conmigo el conocimiento de que en dicho colegio se da una enorme importancia a ese idioma, tanta como a las matemáticas, por ejemplo, y, por tanto, "comprenden" mi afirmación anterior. Todo este tratamiento corresponde fundamentalmente al campo de la Inteligencia Artificial, cuyo desarrollo está avanzando a pasos agigantados.

2. ¿Apoyo al software o al hardware? La máquina de los años venideros, desde luego, deberá tener otro tipo de software más sencillo. El problema del software nunca ha sido resuelto de forma definitiva. La realización de un software bueno requiere mucho trabajo y ensayos, y los cambios que se realizan también resultan muy caros, por lo laboriosos. Además, con frecuencia se presenta la dualidad de:

a) Realizar los cambios necesarios sobre el software existente, lo que en ocasiones supone una pequeña "chapuza", ya que el programa de que se trate era bueno para la situación anterior, pero no para ésta, y debe diseñarse de otro modo.

b) Echar por la borda un enorme trabajo y dinero invertidos en la operación anterior y diseñar el programa de nuevo. Y no hablemos de las adaptaciones entre unas máquinas y otras. El problema es enorme, y resulta muy claro que no puede detenerse el proceso evolutivo para "aprovechar" inversiones pasadas.

Por tanto, el software debe ser diferente. Se puede pensar en que su función se realice a través de algún tipo de hardware.

3. Los ordenadores actuales, sin embargo, han experimentado un enorme auge debido a las técnicas de integración. Estas técnicas todavía se pueden mejorar y aumentar, de forma que se cree un ordenador mucho más potente, formado por muchos chips (verdaderos ordenadores) interconectados. Estos ordenadores deberán ser, además, mucho más rápidos para que puedan procesar enormes cantidades de información. Si lo que precisamos son máquinas "inteligentes", necesitaremos dotarlas de "ojos", "brazos", "cerebro" para tomar decisiones, y el principal problema a resolver con estas técnicas es la enorme cantidad de información a procesar y la rapidez. (Un sistema para aumentar la rapidez es el proceso de la información en paralelo y otro el proceso pipeline. De ellos hablamos en el Glosario.)

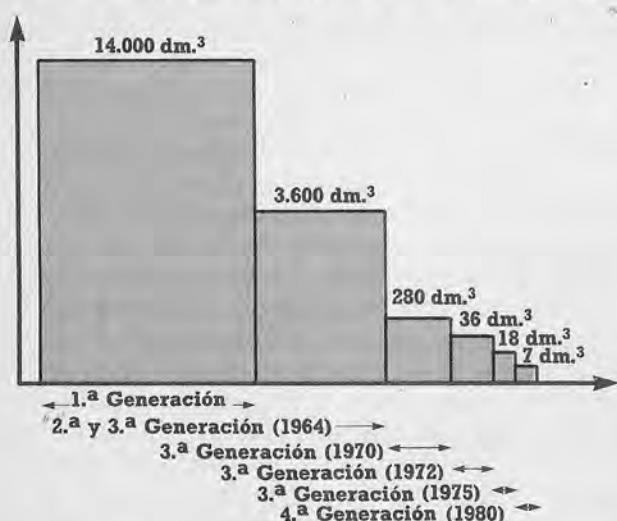


Fig. 2. Volumen ocupado por un millón de bytes de memoria (1 Mb).

Profundicemos algo más en las consideraciones anteriores, y observaremos que los ordenadores actuales realizan operaciones de cálculo secuencial en unos tiempos pequeñísimos. Donde aparecen siempre los problemas es en el tratamiento de la "información". Esto es natural; la información, para ser procesada, debe reducirse a unos códigos binarios o hexadecimales que la máquina procesará como si de dígitos se tratara, volviendo a deshacer los pasos para dar la solución. Evidentemente, esta estructura de tratamiento de la información no es, en absoluto, la adecuada para eso precisamente, para procesar la información. Los ordenadores modernos son todos ellos descendientes del primer "neandertal" que fue el ENIAC de Von Neumann. Y éste y sus predecesores eran máquinas de cálculo, sofisticadas (recuerde el lector los trabajos para codificar los datos de la oficina del censo), que hacían posible procesar la información, aunque para ello hubiera que codificarla.

El ordenador de la década siguiente debe ser una máquina muy superior. Debe ser capaz de realizar deducciones lógicas sofisticadas, y tomar decisiones, o al menos ofrecer posibles decisiones acertadas. Deberán servir para todo tipo de experimentación de laboratorio, economizando muchos recursos, y acelerando la investigación. Para ello, además de su potencia, estarán conectados a enormes bases de datos a nivel mundial, de donde obtendrán los conocimientos necesarios, el "saber", para poder tomar las decisiones.

Hemos visto que el ordenador de esa década será grande y potente, unido a enormes bases de datos a escala mundial, y utili-

zable para simulaciones, entre ellas, la experimentación de laboratorio. Pero ¿no habrá también pequeños ordenadores? Desde luego, los habrá, y aplicados a multitud de facetas de nuestra vida. Algunos serán máquinas "dedicadas", es decir, preparadas para realizar una determinada tarea, aplicables a multitud de cosas, coches, juguetes, control de determinadas labores del hogar, control de electrodomésticos, etc. También habrá muchos pequeños ordenadores conectados a otros más grandes formando enormes redes de comunicación. Como antes hemos indicado, las bases de datos jugarán un papel importantísimo, estando al alcance de cualquier usuario que desee procesar esos conocimientos necesarios para la adecuada toma de decisiones.

¿Serán las máquinas capaces de procesar "conocimientos"? Así lo esperan los japoneses. Los trabajos de Inteligencia Artificial han avanzado muchísimo. Los datos se analizarán considerando también sus interrelaciones y correlaciones, para poder comprender el "contexto".

No sabemos si los japoneses podrán llevar a cabo una labor tan ingente. Muchos investigadores extranjeros los consideran únicamente unos "copiadores" excelentes, pero sin ninguna creatividad.

Sin embargo, para comprender por qué los nipones han emprendido semejante singladura, hay que estudiar su forma de ser, los organismos estatales que han intervenido en esta formidable apuesta, que no tienen por qué parecerse en absoluto a los occidentales.

Por ejemplo, el padre del proyecto, el Ministerio de Comercio Internacional e Industria (MITI), tiene una estructura muy diferente a los ministerios españoles que todos conocemos. Un punto importante es que en España, al sustituir a un ministro, salen con él, desde luego, el jefe de su Gabinete Técnico y la mayoría de los directores generales. Esta vida efímera de los altos cargos es muy perjudicial para la toma de las decisiones adecuadas. Es difícil que una persona piense a algo más de medio plazo, si las decisiones no afectan su supervivencia en el cargo. Además, no le servirá, en absoluto, para hacer méritos con vistas al nuevo cargo. Los funcionarios japoneses del MITI son técnicos muy especializados. Están en cierto modo (nada es perfecto) por encima de la política, ya que saben que su puesto es vitalicio (naturalmente, si lo desempeñan con profesionalidad). Además, al entrar en el organismo van rotando por sus distintos depar-

tamentos, para conocer a fondo la Institución. Todas estas circunstancias, unidas a la propia forma de ser del japonés, muy responsable, y acostumbrado a estructuras paternalistas, hace que se sientan solidarios del Organismo, aceptando las críticas y logros de sus departamentos, como críticas y logros propios.

Como el Japón es, además, un país pequeño y muy poblado, que depende mucho de la exportación, es evidente que los funcionarios del MITI son funcionarios de élite, muy responsables, y acostumbrados a planificar a largo plazo.

Volviendo al proyecto; su presupuesto es enorme. Sin embargo, esta palabra es ambigua. (Las asignaciones del Estado americano para este tipo de investigaciones son mucho mayores.) En principio, el proyecto cuenta con 450 millones de dólares para el decenio, contando con que se irán gastando 45 millones de dólares en los primeros años, para aumentar más adelante los gastos en desarrollo e investigación. Además, el MITI cuenta con que las distintas naciones que colaboran en el proyecto realicen aportaciones de cantidades semejantes. También espera, basándose en los logros obtenidos, conseguir más dinero del Gobierno.

A pesar de esa enorme cuantía, empresas absolutamente comerciales, como IBM, dedican cantidades parecidas e incluso muy superiores en períodos de tiempo menores, sin presentar objetivos tan ambiciosos.

La primera fase del proyecto Quinta Generación será sufragada íntegramente por el MITI. Para el Japón, este proyecto es de una importancia enorme. Sin embargo, a pesar del interés de otras naciones y empresas por esta andadura, el esfuerzo económico es conside-

rable, y, a nivel empresarial, el riesgo que corren es enorme. Los países, por otro lado, siguen con un chauvinismo oculto, que emerge en ocasiones, impidiendo una compenetración mayor.

Los japoneses, desde luego, están empeñados en cambiar la idea occidental de que sólo saben "calcar" a la perfección. Están decididos a mostrar al mundo cómo pueden innovar las tecnologías más avanzadas, sin necesidad de seguir los pasos trillados de otros investigadores extranjeros.

Además, otro de sus objetivos es, ampliar los campos de utilización de los ordenadores. Realmente creen que pueden ayudar a mejorar el nivel de vida, aprovechando mucho mejor los recursos. Existen campos primarios, como la agricultura, en los que apenas si se han utilizado. También son, y serán, de mucha más utilidad estos nuevos ordenadores para tratar problemas sociales, en general problemas que no sean específicos de máquinas de calcular, como antes indicamos. Por ejemplo, los nuevos ordenadores podrán gestionar mejor una oficina de empleo, considerando aptitudes, edad, posibilidades de trabajar sólo unas horas, trabajos para minusválidos, ancianos, trabajo en casa, tipos de trabajos que se deben fomentar, etc., pero todo ello considerado no como información al usuario, sino como una gestión de mejora de las condiciones de empleo y de óptima captación de recursos.

En el siguiente capítulo hablaremos de las reacciones de las distintas naciones y de la formación del equipo responsable. También de los detractores y los posibles fallos del proyecto. Veremos los lenguajes que se desarrollarán y hablaremos de espionaje industrial.

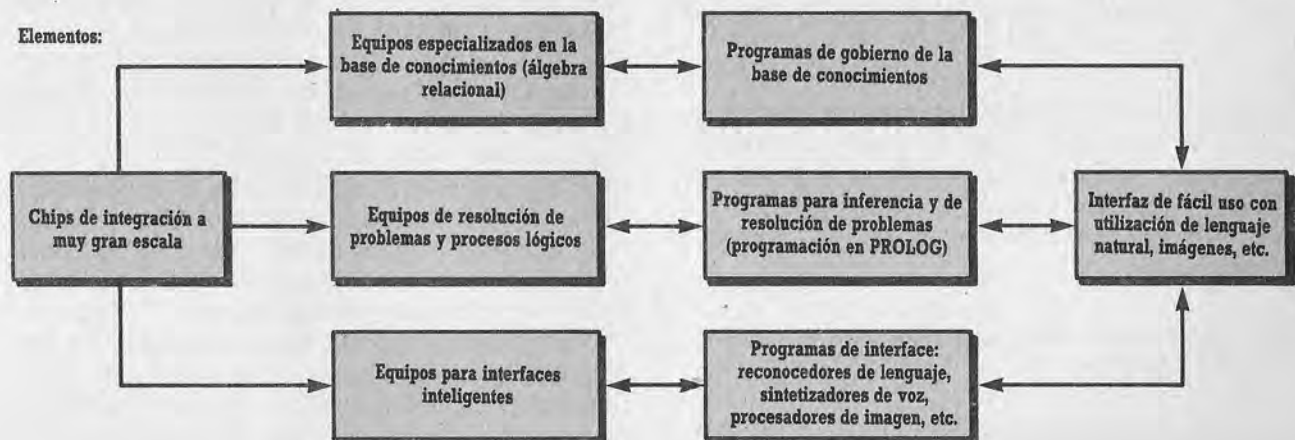


Fig. 3. Estructura de los elementos informáticos en los sistemas de la quinta generación.

TERMINOLOGIA

GLOSARIO DE TERMINOS UTILIZADOS EN RELACION CON EL PROYECTO JAPONES DE QUINTA GENERACION

Base de conocimientos. Conjunto de hipótesis, creencias y hechos sobre los que se basan los restantes elementos del sistema para obtener un diagnóstico o una interpretación.

Base de datos. (Véase **Vocabulario, Datos, Base de.**)

Base de datos relacional. Base de datos en la que se almacenan, además de todos los datos organizados, las relaciones existentes entre ellos para una mayor funcionalidad.

Dirección de la base de conocimientos. Uno de los tres subsistemas del sistema experto. Su misión es organizar, controlar y dirigir la base de conocimientos, actualizándolos inmediatamente. Cuando el motor de inferencia está trabajando en una línea de razonamiento, la dirección busca aquellos conocimientos que puedan incidir en esa línea.

Heurística. Conocimiento experimental y de discernimiento. Conjunto de reglas que maneja una persona experimentada en la materia. Normalmente los resultados que se obtienen son válidos, pero no siempre ocurre así.

IA. Siglas utilizadas para referirse a la Inteligencia Artificial.

Inferencia simbólica. Proceso por el cual a partir de unas premisas se puede, median-

te un razonamiento, llegar a un resultado. El sistema de inferencia trabaja con los datos de la base de conocimientos.

Ingeniería del haber. Ciencia que enseña a diseñar y construir sistemas expertos y otros sistemas de IA.

Integración a muy gran escala. (VLSI- Very Large Scale Integration.) Sistema por el cual en una pequeña pastilla de silicio se colocan multitud de circuitos con transistores, etc. Existen varios grados de integración, pero la integración a muy gran escala incluye en un pequeño chip de silicio unos 500.000 transistores. En el proyecto de Quinta Generación japonés están previstas integraciones de hasta 10.000.000 de elementos.

Interfaz humana. Subsistema de un sistema experto por el cual una persona cualquiera se comunica con la máquina empleando un lenguaje natural, es decir, semejante al utilizado en comunicarse con personas humanas. Una persona debe poder comunicarse con la máquina utilizando su propio lenguaje, a una velocidad normal, y visualizando imágenes enviadas por ésta. Evidentemente, para la comunicación es necesario que la máquina "comprenda" las instrucciones recibidas en un tiempo aceptable. Hasta el momento se han estudiado mucho estos temas (visión artificial, reconocimiento de palabra, etc.), pero se esperan alcanzar cotas mucho más altas en el programa Quinta Generación. Constituye uno de los puntos básicos de estudio.

Proceso en paralelo. (SIMD-Single Instruction Stream, Multiple Data Stream.) Uno de los sistemas pensados para dotar al ordenador de una mayor rapidez en el proceso de la información. Normalmente se utiliza cuando se desea procesar muchísima información y el tiempo es un factor fundamental. El sistema consiste en disponer muchos procesadores en una matriz de dos dimensiones. La información que se va a procesar se envía en partes, simultáneamente a los procesadores, de forma que éstos tratan en cada momento gran número de datos. Los procesadores utilizados se suelen llamar procesadores en matriz. Normalmente estas máquinas son muy difíciles de programar, y no resultan muy útiles. Se utilizan, como antes dijimos, en aquellas ocasiones en las que es imprescindible la rapidez en los cálculos. Por ejemplo, en visión artificial y en investigación meteorológica y sísmica. Los Estados Unidos (en particular la Marina) invirtieron dinero en investigación de este tipo, pero los resultados no han sido muy brillantes.

Proceso en pipeline. (Pipeline significa conducción, tubería.) Otro sistema de proceso de la información que se utiliza también cuando la velocidad es un factor esencial en el proceso. La estructura es absolutamente diferente del proceso en paralelo. En este tipo de proceso la tarea principal se subdivide en subtareas, que se van reali-

zando secuencialmente por los distintos procesadores especializados. (Puede ponerse como ejemplo una cadena de montaje de una fábrica de coches, en la que cada operario va añadiendo algo al montaje, pero está siempre trabajando, de forma que el proceso puede considerarse en cierto modo como un proceso "paralelo".) Este sistema es más eficaz, por el momento, que el proceso en paralelo. La Compañía Texas Instruments, entre otras, ha desarrollado ordenadores que llevan varios sistemas de proceso en pipeline.

Representación del saber. Estructuración de los conocimientos, de forma que puedan ser manipulados por el sistema director de la base de conocimientos.

Sistema experto. Programa que realiza una tarea especializada normalmente basándose en grandes masas de conocimientos adecuadamente organizados, y obtenidos de especialistas. Sirve precisamente para ayudar a estos u otros especialistas en sus tomas de decisiones.

Tratamiento informático del saber. En el proyecto de Quinta Generación los ordenadores podrán procesar el "saber" humano. Para ello utilizarán enormes bases de datos relacionales, interfaces humanas muy rápidas y depuradas, y realizarán inferencias simbólicas.

VOCABULARIO DE INFORMATICA

Conversión. Cambio realizado en la representación de los datos. Por ejemplo, se pueden convertir datos numéricos de forma binaria a decimal, de tarjetas a cinta, etc.

Convex, programación. Caso especial de programación no lineal en la que la función que se va a maximizar o minimizar y los "constraints" son funciones cóncavas o convexas de variables controlables. En contraposición con programación dinámica, lineal, matemática o cuadrática.

Copiar. Reproducir los datos en una nueva posición o en otro destino cualquiera, dejando los datos fuente sin cambios, aunque la forma del resultado pueda ser diferente de la de la fuente. Como ejemplo, podemos indicar la copia de un bloque de tarjetas perforadas en una cinta, la copia de un diskette en el disco duro, etc.

Correctivo, mantenimiento. (Véase **Mantenimiento correctivo**.)

Correctivo, tiempo de mantenimiento. Véase **Tiempo de mantenimiento correctivo**.)

Contador. Dispositivo que se utiliza para representar el número de veces que se ha producido un acontecimiento. Puede ser un registro o una posición de almacenamiento.

CPD. Siglas de Centro de Proceso de Datos. Local en el que se encuentra instalado el or-

denador y donde se llevan a cabo las operaciones necesarias para su adecuado funcionamiento.

CPL. Abreviatura de caracteres por línea. Suele utilizarse para indicar el número de caracteres por línea que puede escribir una impresora. Normalmente suelen ser 80, 120 y 132.

CPU. (Véase **Unidad Central de Proceso**.)

CR. Se utiliza para indicar el carácter de vuelta del carro.

Criogenia. Estudio y uso de dispositivos que utilizan propiedades de los materiales a temperaturas próximas al cero absoluto.

Criotron. Dispositivo que utiliza el efecto causado por temperaturas muy bajas sobre materiales conductores (por ejemplo, pueden utilizarse pequeños cambios en campos magnéticos para controlar corrientes de cuantía muy superior).

CRT, pantalla. Dispositivo de presentación de datos electrostático que utiliza un haz de rayos catódicos como sistema de representación de éstos.

Cursor. Pequeña marca que aparece en la pantalla de los sistemas de presentación visual para indicar la posición donde se escribirá el siguiente carácter que se introduzca a través del teclado, o desde otro dispositivo.

